EXCHNG

```
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCC    000000   PPPPPPPP   YY      YY
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCC    000000   PPPPPPPP   YY      YY
EE           XX    XX    CC          CC        00    00   PP     PP  YY      YY
EE           XX    XX    CC          CC        00    00   PP     PP  YY      YY
EE            XX  XX     CC          CC        00    00   PP     PP   YY    YY
EE            XX  XX     CC          CC        00    00   PP     PP   YY    YY
EEEEEEEE       XX       CC          CC        00    00   PPPPPPPP      YY
EEEEEEEE       XX       CC          CC        00    00   PPPPPPPP      YY
EE            XX  XX     CC          CC        00    00   PP           YY
EE            XX  XX     CC          CC        00    00   PP           YY
EE           XX    XX   CC          CC        00    00   PP           YY          ....
EE           XX    XX   CC          CC        00    00   PP           YY          ....
EEEEEEEEEE  XX      XX   CCCCCCCC    CCCCCCC    000000   PP           YY          ....
EEEEEEEEEE  XX      XX   CCCCCCCC    CCCCCCC    000000   PP           YY          ....

LL           IIIIII    SSSSSSSS
LL           IIIIII    SSSSSSSS
LL             II     SS
LL             II     SS
LL             II     SS
LL             II     SS
LL             II      SSSSSS
LL             II      SSSSSS
LL             II          SS
LL             II          SS
LL             II          SS
LL             II          SS
LLLLLLLLLL   IIIIII    SSSSSSSS
LLLLLLLLLL   IIIIII    SSSSSSSS
```

```
    1    0001  O MODULE   exch$copy                                    %TITLE 'copy verb dispatch and misc routines'
    2    0002  0              (
    3    0003  0              IDENT = 'V04-000',
    4    0004  0              ADDRESSING_MODE (EXTERNAL=LONG_RELATIVE, NONEXTERNAL=WORD_RELATIVE)
    5    0005  0              ) =
    6    0006  1 BEGIN
    7    0007  1 !
    8    0008  1 !********************************************************************************
    9    0009  1 !*                                                                              *
   10    0010  1 !*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                    *
   11    0011  1 !*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                     *
   12    0012  1 !*   ALL RIGHTS RESERVED.                                                       *
   13    0013  1 !*                                                                              *
   14    0014  1 !*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED      *
   15    0015  1 !*   ONLY IN ACCORDANCE WITH THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
   16    0016  1 !*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER      *
   17    0017  1 !*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY      *
   18    0018  1 !*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY      *
   19    0019  1 !*   TRANSFERRED.                                                               *
   20    0020  1 !*                                                                              *
   21    0021  1 !*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE      *
   22    0022  1 !*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT      *
   23    0023  1 !*   CORPORATION.                                                               *
   24    0024  1 !*                                                                              *
   25    0025  1 !*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS      *
   26    0026  1 !*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                    *
   27    0027  1 !*                                                                              *
   28    0028  1 !*                                                                              *
   29    0029  1 !********************************************************************************
   30    0030  1 !
   31    0031  1 !++
   32    0032  1 ! FACILITY:      EXCHANGE - Foreign volume interchange facility
   33    0033  1 !
   34    0034  1 ! ABSTRACT:      Primary action routines for copy verb
   35    0035  1 !
   36    0036  1 ! ENVIRONMENT:   VAX/VMS User mode
   37    0037  1 !
   38    0038  1 ! AUTHOR:        CW Hobbs          CREATION DATE: 2-Oct-1982
   39    0039  1 !
   40    0040  1 ! MODIFIED BY:
   41    0041  1 !
   42    0042  1 !       V03-002 CWH3002                 CW Hobbs                12-Apr-84
   43    0043  1 !               Return file-not-found correctly in copy loop. Modify TYPE
   44    0044  1 !               output routine for eight-bit character set.
   45    0045  1 !
   46    0046  1 !
   47    0047  1 !--
   48    0048  1 !
   49    0049  1 ! Include files:
   50    0050  1 !
   51    0051  1 MACRO $module_name_string = 'exch$copy' %;           ! The require file needs to know our module name
   52    0052  1 REQUIRE 'SRC$:EXCREQ'                                ! Facility-wide require file
   53    0053  1     ;
```

EXCH$COPY   copy verb dispatch and misc routines   E 16   VAX-11 Bliss-32 V4.0-742   Page 2
V04-000   Module table of contents   16-Sep-1984 00:41:48   (2)
                                                            5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1

```
      55      0150  1  %SBTTL 'Module table of contents'
      56      0151  1
      57      0152  1  ! Module table of contents:
      58      0153  1
      59      0154  1  FORWARD ROUTINE
      60      0155  1      exch$copy_copy,                              ! Main action routine for COPY verb
      61      0156  1          copy_init              : NOVALUE,        ! Inits common to COPY and TYPE
      62      0157  1          copy_input_close       : NOVALUE,        ! Close the input file
      63      0158  1          copy_input_open                          ! Open the input file
      64      0159  1      exch$copy_namb_to_filb     : NOVALUE,        ! Copy fields from namb to the filb
      65      0160  1          copy_output_cleanup    : NOVALUE,        ! Release structures and clean up output
      66      0161  1          copy_output_close,                       ! Close the output file
      67      0162  1          copy_output_create,                      ! Create the output file
      68      0163  1          copy_output_delete     : NOVALUE,        ! Delete the output file after error
      69      0164  1          copy_parse_cleanup     : NOVALUE,        ! Release structures and clean up after parse
      70      0165  1          copy_parse_next_input,                   ! Fetch and expand next input parameter
      71      0166  1      exch$copy_type,                              ! Main action routine for TYPE verb
      72      0167  1          copy_type_print        : NOVALUE         ! Reformat and print lines on SYS$OUTPUT
      73      0168  1          ;
      74      0169  1
      75      0170  1  ! EXCHANGE facility routines
      76      0171  1  !
      77      0172  1  EXTERNAL ROUTINE
      78      0173  1      exch$cmd_cli_get_integer,                    ! Get an integer value
      79      0174  1      exch$cmd_parse_filespec,                     ! Parse a file specification
      80      0175  1      exch$dos11_create_file,                      ! Create and connect to a DOS-11 file
      81      0176  1      exch$dos11_open_file,                        ! Connect to a DOS-11 file
      82      0177  1      exch$fil11_create_file,                      ! Create and connect to an RMS file
      83      0178  1      exch$fil11_open_file,                        ! Connect to an RMS file
      84      0179  1      exch$moun_implied_mount,                     ! Do a default mount
      85      0180  1      exch$rt11_create_file,                       ! Create and connect to an RT11 file
      86      0181  1      exch$rt11_open_file,                         ! Connect an RT11 file
      87      0182  1      exch$rt11_write_cleanup    : NOVALUE,        ! Complete writing to an RT-11 volume
      88      0183  1      exch$rt11_write_prepare    : NOVALUE,        ! Prepare to write to an RT-11 volume
      89      0184  1      exch$util_dos11ctx_release : NOVALUE,        ! Release dos-11 block
      90      0185  1      exch$util_fao_buffer,                        ! Format an fao string
      91      0186  1      exch$util_filb_allocate,                     ! Allocate file context block
      92      0187  1      exch$util_filb_release     : NOVALUE,        ! Release file context block
      93      0188  1      exch$util_file_error,                        ! Tell about an rms error
      94      0189  1      exch$util_namb_release     : NOVALUE,        ! Release name block
      95      0190  1      exch$util_rmsb_allocate,                     ! Allocate Files-11 control block
      96      0191  1      exch$util_rmsb_release     : NOVALUE,        ! Release Files-11 block
      97      0192  1      exch$util_rt11ctx_allocate,                  ! Allocate RT-11 context block
      98      0193  1      exch$util_rt11ctx_release  : NOVALUE,        ! Release RT-11 block
      99      0194  1      exch$util_vm_allocate                        ! Allocate virtual memory
     100      0195  1          ;
     101      0196  1
     102      0197  1  ! Equated symbols:
     103      0198  1  !
     104      0199  1  !LITERAL
     105      0200  1  !    ;
     106      0201  1
     107      0202  1  ! Bound declarations:
     108      0203  1  !
     109      0204  1  BIND
     110      0205  1      ascid_allocation       = %ASCID 'ALLOCATION',            ! Save some space, these strings used more than once
     111      0206  1      ascid_best_try         = %ASCID 'BEST_TRY_CONTIGUOUS',
```

```
: 112          0207 1     ascid_contiguous     = %ASCID 'CONTIGUOUS',
: 113          0208 1     ascid_extension      = %ASCID 'EXTENSION',
: 114          0209 1     ascid_truncate       = %ASCID 'TRUNCATE'
: 115          0210 1     ;
```

EXCH$COPY   copy verb dispatch and misc routines   G 16   VAX-11 Bliss-32 V4.0-742   Page   4
VO4-000   exch$copy_copy   16-Sep-1984 00:41:48   (3)
                                                                 5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1

```
: 117    0211  1 GLOBAL ROUTINE exch$copy_copy = %SBTTL 'exch$copy_copy'
: 118    0212  2 BEGIN
: 119    0213  2 !++
: 120    0214  2 !
: 121    0215  2 !  FUNCTIONAL DESCRIPTION:
: 122    0216  2 !
: 123    0217  2 !      Action routine for the copy verb, parses and performs main control functions for copy
: 124    0218  2 !
: 125    0219  2 !  INPUTS:
: 126    0220  2 !
: 127    0221  2 !      none
: 128    0222  2 !
: 129    0223  2 !  IMPLICIT INPUTS:
: 130    0224  2 !
: 131    0225  2 !      Command parameters and qualifiers as returned from CLI$ routines.  Global environment ref'd by exch$
: 132    0226  2 !
: 133    0227  2 !  OUTPUTS:
: 134    0228  2 !
: 135    0229  2 !      none
: 136    0230  2 !
: 137    0231  2 !  IMPLICIT OUTPUTS:
: 138    0232  2 !
: 139    0233  2 !      none
: 140    0234  2 !
: 141    0235  2 !  ROUTINE VALUE:
: 142    0236  2 !
: 143    0237  2 !      Success or worst error encountered.
: 144    0238  2 !
: 145    0239  2 !  SIDE EFFECTS:
: 146    0240  2 !
: 147    0241  2 !      Files may be created.
: 148    0242  2 !--
: 149    0243  2
: 150    0244  2 $dbgtrc_prefix ('copy_copy> ');
: 151    0245  2
: 152    0246  2 LOCAL
: 153    0247  2     copy : $ref_bblock,                                      ! Pointer to work area
: 154    0248  2     inp_filb : $ref_bblock,
: 155    0249  2     out_filb : $ref_bblock,
: 156    0250  2     out_namb : $ref_bblock,
: 157    0251  2     abort,
: 158    0252  2     protect,
: 159    0253  2     prs_stat,
: 160    0254  2     status
: 161    0255  2     ;
: 162    0256  2
: 163    0257  2
: 164    0258  2 ! Allocate and/or initialize the work area
: 165    0259  2 !
: 166    0260  2 copy_init ();
: 167    0261  2
: 168    0262  2 ! Get pointers that we need.  Have to wait until work area is allocated by init call
: 169    0263  2 !
: 170    0264  2 copy = .exch$a_gbl [excg$a_copy_work];                ! Pointer to work area
: 171    0265  2
: 172    0266  2 ! Init the name used for the input file default.  As long as it is null we can also use it for output defaul
: 173    0267  2 !
```

```
EXCH$COPY        copy verb dispatch and misc routines          16-Sep-1984 00:41:48    VAX-11 Bliss-32 V4.0-742       Page  5
V04-000          exch$copy_copy                                  5-Sep-1984 22:04:55    [EXCHNG.SRC]EXCCOPY.B32;1            (3)
```

```
H 16

  174    0268    2   str$copy_dx (copy [copy$q_input_sticky_name], %ASCID '');
  175    0269
  176    0270    2   ! Get the string and the namb for the output filename.  By fetching this parameter, we will pick up position
  177    0271    2   ! qualifiers attached to the second parameter.
  178    0272    2   !
  179    0273    3   IF NOT (status = exch$cmd_parse_filespec (%ASCID 'OUTPUT', copy [copy$q_input_sticky_name], 0,
  180    0274                               copy [copy$q_output_filename], out_namb))
  181    0275    2   THEN
  182    0276    2       $exch_signal_return (exch$_parseerr, 1, copy [copy$q_output_filename], .status);
  183    0277    2   $debug_print_fao ('output parameter is "!AS"', copy [copy$q_output_filename]);
  184    0278    2   copy [copy$a_out_namb] = .out_namb;                    ! Save the address of the namb in the work area
  185    0279
  186    0280    2   ! Get the default set of boolean qualifiers, note that we treat positionals on the second parameter as globa
  187    0281    2   !
  188    0282    2   copy [copy$v_q_best_try_contiguous] = cli$present (ascid_best_try);              ! positional
  189    0283    2   copy [copy$v_q_contiguous]          = cli$present (ascid_contiguous);            ! positional
  190    0284    2   copy [copy$v_q_delete]              = cli$present (%ASCID 'DELETE');             ! positional
  191    0285    2   copy [copy$v_q_replace]             = cli$present (%ASCID 'REPLACE');            ! positional
  192    0286    2   copy [copy$v_q_system]              = cli$present (%ASCID 'SYSTEM');             ! global
  193    0287    2   copy [copy$v_q_truncate]            = cli$present (ascid_truncate);              ! positional
  194    0288    2
  195    0289    2   ! For /PROTECT, we need to know whether it was specified or defaulted
  196    0290    2   !
  197    0291    2   protect = cli$present (%ASCID 'PROTECT');
  198    0292    2   copy [copy$v_q_protect]          = .protect;                                     ! Simply value of low bit
  199    0293    4   copy [copy$v_q_protect_explicit] = ((.protect EQL cli$_present)                  ! Either /PROTECT or /NOPROT
  200    0294    2                                      OR (.protect EQL cli$_negated));             !  must be there
  201    0295    2
  202    0296    2   ! Get individual integer-valued qualifiers, routine signals on errors.  If the qualifier is not present, 0 i
  203    0297    2   ! in the second parameter and -1 (success) is returned as the routine value.  Here we also treat positionals
  204    0298    2   ! second parameter as globals.
  205    0299    2   !
  206    0300    3   IF NOT (status = exch$cmd_cli_get_integer (ascid_allocation, copy [copy$l_q_allocation]))
  207    0301    2   THEN
  208    0302    3       BEGIN
  209    0303    3       exch$util_namb_release (.out_namb);
  210    0304    3       RETURN .status;
  211    0305    2       END;
  212    0306    2
  213    0307    3   IF NOT (status = exch$cmd_cli_get_integer (ascid_extension, copy [copy$l_q_extension]))
  214    0308    2   THEN
  215    0309    3       BEGIN
  216    0310    3       exch$util_namb_release (.out_namb);
  217    0311    3       RETURN .status;
  218    0312    2       END;
  219    0313    2
  220    0314    3   IF NOT (status = exch$cmd_cli_get_integer (%ASCID 'START_BLOCK', copy [copy$l_q_start_block]))
  221    0315    2   THEN
  222    0316    3       BEGIN
  223    0317    3       exch$util_namb_release (.out_namb);
  224    0318    3       RETURN .status;
  225    0319    2       END;
  226    0320    2
  227    0321    2   ! If a foreign device is not mounted, then perform an implied mount
  228    0322    2   !
  229    0323    3   IF   (.out_namb [namb$a_assoc_volb] EQL 0)
  230    0324    2     AND
```

EXCH$COPY   copy verb dispatch and misc routines   I 16
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 6
                                                                       5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (3)

```
231    0325  4          (BEGIN
232    0326  4          BIND
233    0327  4            dev = out_namb [namb$l_fabdev] : $bblock;
234    0328  5          .dev [dev$v_for] OR (NOT (.dev [dev$v_mnt]))
235    0329  5          END)
236    0330  2       AND
237    0331  4          ((.out_namb [namb$b_devclass] EQL dc$_disk)
238    0332  3          OR
239    0333  3          (.out_namb [namb$b_devclass] EQL dc$_tape))
240    0334  2    THEN
241    0335  3        BEGIN
242    0336
243    0337  4        IF NOT (status = exch$moun_implied_mount (.out_namb))
244    0338  3        THEN
245    0339  4            BEGIN
246    0340  4            exch$util_namb_release (.out_namb);
247    0341  4            RETURN .status;
248    0342  3            END;
249    0343  2        END;
250    0344  2
251    0345  2   ! If the device has a volb, make sure that the volb is valid and that write access is permitted.
252    0346
253    0347  3   IF (.out_namb [namb$a_assoc_volb] NEQ 0)
254    0348  2   THEN
255    0349  3        BEGIN
256    0350  3        BIND
257    0351  3            volb = out_namb [namb$a_assoc_volb] : $ref_bblock;
258    0352
259    0353  3        ! We should now have a valid volb, but we still should check
260    0354
261    0355  3        $block_check (2, .volb, volb, 496);
262    0356
263    0357  3        ! Make certain that write access is permitted
264    0358
265    0359  3        IF NOT .volb [volb$v_write]
266    0360  3        THEN
267    0361  4            BEGIN
268    0362  4            $exch_signal (exch$_nocoplock, 2, .volb [volb$l_vol_ident_len], volb [volb$t_vol_ident]);
269    0363  4            exch$util_namb_release (.out_namb);
270    0364  4            RETURN exch$_nocoplock;
271    0365  3            END;
272    0366
273    0367  3        CASE .volb [volb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
274    0368  3            SET
275    0369
276    0370  3            [volb$k_vfmt_rt11] :
277    0371  4                            BEGIN
278    0372  4                            IF .out_namb [namb$v_bad_pdp_char]
279    0373  4                                OR
280    0374  4                                .out_namb [namb$v_rt_truncate]
281    0375  4                            THEN
282    0376  5                                BEGIN
283  P 0377  5                                $exch_signal (exch$_badfilename, 3, out_namb [namb$q_input]
284    0378  5                                              .volb [volb$l_vol_type_len], volb [volb$t_vol_type]);
285    0379  5                                exch$util_namb_release (.out_namb);
286    0380  5                                RETURN exch$_badfilename;
287    0381  4                                END;
```

EXCH$COPY   copy verb dispatch and misc routines   J 16   VAX-11 Bliss-32 V4.0-742   Page 7
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   [EXCHNG.SRC]EXCCOPY.B32;1   (3)
                                                                            5-Sep-1984 22:04:55

```
:  288        0382   4                                      exch$rt11_write_prepare (.volb);                    ! Do sundries necessary before we start copy
:  289        0383   3                                   END;
:  290        0384   3
:  291        0385   3                    [volb$k_vfmt_dos11] :
:  292        0386   4                                   BEGIN
:  293        0387   4                                   IF .out_namb [namb$v_bad_pdp_char]
:  294        0388   4                                      OR
:  295        0389   4                                      .out_namb [namb$v_dos_truncate]
:  296        0390   4                                   THEN
:  297        0391   5                                      BEGIN
:  298      P 0392   5                                      $exch_signal (exch$_badfilename, 3, out_namb [namb$q_input],
:  299        0393   5                                                         .volb [volb$l_vol_type_len], volb [volb$t_vol_type]);
:  300        0394   5                                      exch$util_namb_release (.out_namb);
:  301        0395   5                                      RETURN exch$_badfilename;
:  302        0396   4                                      END;
:  303        0397   3                                   END;
:  304        0398   3
:  305        0399   3                    [INRANGE, OUTRANGE] :
:  306        0400   3                                   ;                                                    ! Nothing to do for these guys
:  307        0401   3                    TES;
:  308        0402   2              END;
:  309        0403   2
:  310        0404   2  ! Allocate a file block to contain the output file information
:  311        0405   2  !
:  312        0406   2  out_filb = exch$util_filb_allocate ();
:  313        0407   2  copy [copy$a_out_filb] = .out_filb;                      ! Save the address of the filb in the work area
:  314        0408   2  exch$copy_namb_to_filb (.out_namb, .out_filb);           ! Move some data from the namb to the filb
:  315        0409   2
:  316        0410   2
:  317        0411   2  ! Loop through the list of input file specifications.  Errors will be signalled.  If an error occurs the cur
:  318        0412   2  ! input element is skipped and processing continues with the next input item.
:  319        0413   2  !
:  320        0414   2  abort = false;
:  321        0415   2  status = ss$_normal;
:  322        0416   3  WHILE (prs_stat = copy_parse_next_input ())               ! Get next input file parameter
:  323        0417   2  DO
:  324        0418   3      BEGIN
:  325        0419   3      LOCAL
:  326        0420   3          ino_stat;
:  327        0421   3
:  328        0422   3      inp_filb = .copy [copy$a_inp_filb];                   ! Grab the pointer to the input filb
:  329        0423   3
:  330        0424   3      ! Check for some invalid naming conditions
:  331        0425   3      !
:  332        0426   3      IF .copy [copy$v_multiple_files]                      ! If the input could map multiple files
:  333        0427   3      THEN
:  334        0428   4          BEGIN
:  335        0429   4
:  336        0430   4          ! Complain if the output file name is explicitly a single file name
:  337        0431   4          !
:  338        0432   6          IF NOT (      (.out_namb [namb$v_wildcard])                              ! A wildcard will help us out
:  339        0433   5                    OR
:  340        0434   6                    (NOT .out_namb [namb$v_explicit_name])                         ! A missing name will work
:  341        0435   5                    OR
:  342        0436   6                    (NOT .out_namb [namb$v_explicit_type])                         ! A missing type can also map multip
:  343        0437   5                    )
:  344        0438   4              THEN
```

EXCH$COPY   copy verb dispatch and misc routines   K 16
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page   8
                                                                  5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (3)

```
:  345       0439  5                      BEGIN
:  346       0440  5                      status = exch$_many_to_one;
:  347       0441  5                      $exch_signal (.status);
:  348       0442  5                      copy_parse_cleanup ();
:  349       0443  5                      EXITLOOP;
:  350       0444  4                      END;
:  351       0445  4
:  352       0446  4            ! Also complain if /START_BLOCK has been requested, since it is hard to put several files on the sam
:  353       0447  4            !
:  354       0448  4            IF .copy [copy$l_q_start_block] NEQ 0
:  355       0449  4            THEN
:  356       0450  5                      BEGIN
:  357       0451  5                      status = exch$_strtnomulti;
:  358       0452  5                      $exch_signal (.status);
:  359       0453  5                      copy_parse_cleanup ();
:  360       0454  5                      EXITLOOP;
:  361       0455  4                      END;
:  362       0456  3            END;
```

EXCH$COPY   copy verb dispatch and misc routines   L 16
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page   9
                                                                  5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (4)

```
 364    0457  3              WHILE 1
 365    0458  3              DO
 366    0459  4                  BEGIN
 367    0460  4
 368    0461  4                      ! If a control/c is pending, don't bother with opening another file
 369    0462  4                      !
 370    0463  4                      IF .exch$a_gbl [excg$v_control_c]
 371    0464  4                      THEN
 372    0465  5                          BEGIN
 373    0466  5                          ino_stat = exch$_canceled;
 374    0467  5                          $exch_signal ($info_stat_copy (.ino_stat));
 375    0468  5                          END
 376    0469  4                      ELSE
 377    0470  4                          ino_stat = copy_input_open ();      ! Open the input file, loop for wildcards
 378    0471  4
 379    0472  4                      ! Remember if this is a reopen, and clear the reopen flag
 380    0473  4                      !
 381    0474  4                      copy [copy$v_reopen_in_progress] = .copy [copy$v_reopen_input];
 382    0475  4                      copy [copy$v_reopen_input] = false;     ! Clear any possible retry
 383    0476  4
 384    0477  4                      IF .ino_stat
 385    0478  4                      THEN
 386    0479  5                          BEGIN
 387    0480  5                          LOCAL
 388    0481  5                              cre_stat,
 389    0482  5                              rec_count;
 390    0483  5
 391    0484  5                          ! Now create the file and copy the records
 392    0485  5                          !
 393    0486  6                          IF (cre_stat = copy_output_create ())       ! Open the output file
 394    0487  5                          THEN
 395    0488  6                              BEGIN
 396    0489  6                              LOCAL
 397    0490  6                                  getput_err,
 398    0491  6                                  cop_stat,
 399    0492  6                                  get_stat,
 400    0493  6                                  put_stat;
 401    0494  6
 402    0495  6                              ! While we can get records move them to the output
 403    0496  6                              !
 404    0497  6                              rec_count = put_stat = getput_err = 0;
 405    0498  7                              WHILE (get_stat = (.inp_filb [filb$a_get_routine]) (.inp_filb))
 406    0499  6                              DO
 407    0500  7                                  BEGIN
 408    0501  7                                  IF NOT (put_stat = (.out_filb [filb$a_put_routine]) ()) THEN EXITLOOP;
 409    0502  7                                  rec_count = .rec_count + 1;
 410    0503  7
 411    0504  7                                  ! If we have seen control/c, exit the loop with a canceled error
 412    0505  7                                  !
 413    0506  7                                  IF .exch$a_gbl [excg$v_control_c]
 414    0507  7                                  THEN
 415    0508  8                                      BEGIN
 416    0509  8                                      put_stat = exch$_canceled;
 417    0510  8                                      abort = true;
 418    0511  8                                      $exch_signal ($info_stat_copy (.put_stat));
 419    0512  8                                      EXITLOOP;
 420    0513  7                                      END;
```

EXCHSCOPY   copy verb dispatch and misc routines   M 16   VAX-11 Bliss-32 V4.0-742   Page 10
VO4-000   exch$copy_copy   16-Sep-1984 00:41:48   [EXCHNG.SRC]EXCCOPY.B32;1   (4)
                                                                  5-Sep-1984 22:04:55

```
  421   0514  6                                      END;
  422   0515  6
  423   0516  6                                      $trace_print_fao ('status !XL, get_stat !XL, put_stat !XL', .status, .get_stat, .put_stat);
  424   0517  6
  425   0518  7                                      IF (NOT .get_stat) AND (.get_stat NEQ 0)
  426   0519  6                                      THEN
  427   0520  7                                          BEGIN
  428   0521  7                                          status = .get_stat;
  429   0522  7                                          getput_err = true;
  430   0523  6                                          END;
  431   0524  6
  432   0525  7                                      IF (NOT .put_stat) AND (.put_stat NEQ 0)
  433   0526  6                                      THEN
  434   0527  7                                          BEGIN
  435   0528  7                                          status = .put_stat;
  436   0529  7                                          getput_err = true;
  437   0530  6                                          END;
  438   0531  6
  439   0532  6                                      ! If we have an error before any records are transferred, try to delete the file
  440   0533  6                                      !
  441   0534  6                                      IF NOT .out_filb [filb$v_file_erased]   ! Output file is still valid
  442   0535  6                                      THEN
  443   0536  7                                          BEGIN
  444   0537  7                                          IF  .getput_err                              ! If we had an error during our get or put a
  445   0538  7                                             AND                                       ! either have transferred no records or we
  446   0539  9                                             ((.rec_count EQL 0)                       ! delete a previous version during close.
  447   0540  8                                              OR                                       ! not want to delete the previous copy of t
  448   0541  9                                              (.out_filb [filb$v_delete_previous])     ! if there was an error in the transfer.
  449   0542  8                                              OR                                       ! We also want to delete the output file if
  450   0543  8                                              (.exch$a_gbl [excg$v_control_c]))        ! command was canceled.
  451   0544  7                                          THEN
  452   0545  8                                              BEGIN
  453   0546  8                                              copy_output_delete ();                   ! Delete the output file if supported for th
  454   0547  8                                              rec_count = 0;                           ! Make sure we get 'NOTCOPIED' message
  455   0548  8                                              END
  456   0549  8
  457   0550  8                                          ! Close the ouput file
  458   0551  8                                          !
  459   0552  7                                          ELSE
  460   0553  8                                              BEGIN
  461   0554  8                                              LOCAL
  462   0555  8                                                  cls_stat;
  463   0556  8                                              cls_stat = copy_output_close ();         ! Close the output file, clean up
  464   0557  8                                              $trace_print_fao ('status !XL, cls_stat !XL', .status, .cls_stat);
  465   0558  8                                              IF NOT .cls_stat
  466   0559  8                                              THEN
  467   0560  9                                                  BEGIN
  468   0561  9                                                  status = .cls_stat;
  469   0562  9                                                  getput_err = true;
  470   0563  8                                                  END;
  471   0564  7                                              END;
  472   0565  6                                          END;
  473   0566  6
  474   0567  6                                      ! If the file has been erased, set record count to zero.  The file might have been erased be
  475   0568  6                                      ! of an I/O error during close, therefore we must do this here.
  476   0569  6                                      !
  477   0570  6                                      IF .out_filb [filb$v_file_erased]
```

EXCHSCOPY   copy verb dispatch and misc routines   B 1
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 11
                                                         5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (4)

```
478    0571  6                              THEN
479    0572  7                                  BEGIN
480    0573  7                                  rec_count = 0;
481    0574  7                                  out_filb [filb$v_file_erased] = false;
482    0575  6                                  END;
483    0576  6
484    0577  6                              ! Set the cop_stat if we need to signal
485    0578  6                              !
486    0579  6                              cop_stat = 0;                              ! Start by assuming no signal
487    0580  6                              IF .getput_err                            ! We had an error which might have caused a partial copy
488    0581  6                              THEN
489    0582  7                                  BEGIN
490    0583  7                                  IF .rec_count EQL 0           ! No recs will get the NOTCOPIED message
491    0584  7                                  THEN
492    0585  7                                      cop_stat = exch$_notcopied
493    0586  7                                  ELSE
494    0587  7                                      cop_stat = exch$_partcopied;
495    0588  7                                  END
496    0589  6                              ELSE
497    0590  7                                  BEGIN
498    0591  7                                  IF  .out_filb [filb$v_name_change]          ! If the name has changed
499    0592  7                                     AND
500    0593  7                                     NOT .copy [copy$v_q_nolog_explicit]      ! But not if /NOLOG was seen
501    0594  7                                  THEN
502    0595  7                                      cop_stat = exch$_copnewname
503    0596  7                                  ELSE IF .copy [copy$v_q_log]                ! /LOG is in effect
504    0597  7                                          OR
505    0598  7                                          .copy [copy$v_reopen_in_progress]   ! Or we have successfully retried the operat
506    0599  7                                  THEN
507    0600  7                                      cop_stat = exch$_copied;
508    0601  6                                  END;
509    0602  6
510    0603  6                              ! If we are going to retry, give that signal
511    0604  6                              !
512    0605  6                              IF .copy [copy$v_reopen_input]
513    0606  6                              THEN
514    0607  6                                  cop_stat = exch$_notcop_retry;
515    0608  6
516    0609  6                              ! If the command was canceled at the keyboard, then do not signal
517    0610  6                              !
518    0611  6                              IF .exch$a_gbl [excg$v_control_c]
519    0612  6                              THEN
520    0613  6                                  cop_stat = 0;
521    0614  6
522    0615  6                              ! Now, if we have a status do the signal
523    0616  6                              !
524    0617  6                              IF .cop_stat NEQ 0
525    0618  6                              THEN
526    0619  7                                  BEGIN
527    0620  7                                  LOCAL
528    0621  7                                      b_or_r;
529    0622 10                                  b_or_r = (IF ((.out_filb [filb$b_rec_format] EQL filb$k_rfmt_fixed)
530    0623  9                                              AND (.out_filb [filb$l_fixed_len] EQL 512))
531    0624  8                                          OR
532    0625  9                                              (.out_filb [filb$b_transfer_mode] EQL filb$k_xfrm_block
533    0626  9                                              OR .inp_filb [filb$b_transfer_mode] EQL filb$k_xfrm_block)
534    0627  7                                      THEN %ASCID 'block' ELSE %ASCID 'record');
```

EXCH$COPY   copy verb dispatch and misc routines   C 1
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 12
                                                         5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (4)

```
 535    P 0628  7                              $exch_signal (.cop_stat, 6,
 536    P 0629  7                                                 .inp_filb [filb$l_result_name_len], inp_filb [filb$t_result_name
 537    P 0630  7                                                 .out_filb [filb$l_result_name_len], out_filb [filb$t_result_name
 538      0631  7                                                 .rec_count, .b_or_r);
 539      0632  6                              END;
 540      0633  6                          END
 541      0634  6
 542      0635  6              ! Able to open input, but not output.  Give the "File not copied" message
 543      0636  6              !
 544      0637  5              ELSE
 545      0638  6                  BEGIN
 546    P 0639  6                  $exch_signal (exch$_notcopied, 4, .inp_filb [filb$l_result_name_len], inp_filb [filb$t_resul
 547      0640  6                                      .out_filb [filb$l_result_name_len], out_filb [filb$t_result_name], .cre_
 548      0641  6                  $trace_print_fao ('status !XL, cre_stat !XL', .status, .cre_stat);
 549      0642  6                  status = .cre_stat;
 550      0643  6
 551      0644  6                  ! Some errors should terminate the command, for example if the directory has overflowed ther
 552      0645  6                  ! no hope of accomplishing anything useful in this command.
 553      0646  6                  !
 554      0647  6                  SELECTONE .cre_stat OF
 555      0648  6                  SET
 556      0649  6                      [0, exch$_rt11_overflow, exch$_volume_full, exch$_volume_full, exch$_nocopsamdev,
 557      0650  6                          exch$_illmtcopy, rms$_dev] :
 558      0651  6                                                    abort = true;
 559      0652  6                      [OTHERWISE] :
 560      0653  6                                                    ;
 561      0654  6                  TES;
 562      0655  5
 563      0656  5                  END;
 564      0657  5
 565      0658  5              copy_input_close ();
 566      0659  5              IF .abort THEN EXITLOOP;
 567      0660  5              END
 568      0661  5
 569      0662  5          ! We got an error from the input_open, but we aren't done yet
 570      0663  5          !
 571      0664  4          ELSE
 572      0665  5              BEGIN
 573      0666  5              $trace_print_fao ('status !XL, ino_stat !XL', .status, .ino_stat);
 574      0667  5
 575      0668  5              IF .ino_stat EQL 0
 576      0669  5                  OR
 577      0670  5                  .exch$a_gbl [excg$v_control_c]
 578      0671  5              THEN
 579      0672  5                  EXITLOOP
 580      0673  5              ELSE
 581      0674  6                  BEGIN
 582      0675  6                  status = .ino_stat;
 583      0676  6                  SELECTONE .ino_stat OF                 ! Some errors call for leaving the loop
 584      0677  6                  SET
 585      0678  6                      [rms$_fnf, rms$_dev] :
 586      0679  6                                      EXITLOOP;
 587      0680  6                      [OTHERWISE] :                      ! Continue to try for all other errors
 588      0681  6                                          ;
 589      0682  5                  TES;
 590      0683  5                  END;
 591      0684  5
```

EXCH$COPY   copy verb dispatch and misc routines   D 1
V04-000   exch$copy_copy   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 13
                                                                 5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (4)

```
592    0685  4              END;
593    0686  3          END;
594    0687  3
595    0688  3      copy_parse_cleanup ();                        ! Release namb, clean up after parse
596    0689  3      IF .abort THEN EXITLOOP;
597    0690  3      END;
598    0691  2
599    0692  2  ! If we had an unusual return from copy_parse_input then use that as the final status
600    0693  2
601    0694  2  $trace_print_fao ('status !XL, prs_stat !XL', .status, .prs_stat);
602    0695  2  IF (NOT .prs_stat) AND (.prs_stat NEQ 0) THEN status = .prs_stat;
603    0696  2
604    0697  2  ! Clean up the structures associated with the output file
605    0698  2  !
606    0699  2  copy_output_cleanup ();
607    0700  2
608    0701  2  $trace_print_fao ('status !XL  (exit)', .status);
609    0702  2  RETURN .status;
610    0703  1  END;
```

```
                                                         .TITLE   EXCH$COPY copy verb dispatch and misc routines
                                                         .IDENT   \V04-000\

                                                         .PSECT   EXCH$COPY_PLIT,NOWRT,2

         00 00 4E 4F 49 54 41 43 4F 4C 4C 41  00000 P.AAB:  .ASCII   \ALLOCATION\<0><0>
                                   010E000A  0000C P.AAA:  .LONG    17694730
                                   00000000' 00010         .ADDRESS P.AAB
47 49 54 4E 4F 43 5F 59 52 54 5F 54 53 45 42  00014 P.AAD:  .ASCII   \BEST_TRY_CONTIGUOUS\<0>
                            00 53 55 4F 55     00023
                                   010E0013  00028 P.AAC:  .LONG    17694739
                                   00000000' 0002C         .ADDRESS P.AAD
         00 00 53 55 4F 55 47 49 54 4E 4F 43  00030 P.AAF:  .ASCII   \CONTIGUOUS\<0><0>
                                   010E000A  0003C P.AAE:  .LONG    17694730
                                   00000000' 00040         .ADDRESS P.AAF
         00 00 00 4E 4F 49 53 4E 45 54 58 45  00044 P.AAH:  .ASCII   \EXTENSION\<0><0><0>
                                   010E0009  00050 P.AAG:  .LONG    17694729
                                   00000000' 00054         .ADDRESS P.AAH
                  45 54 41 43 4E 55 52 54     00058 P.AAJ:  .ASCII   \TRUNCATE\
                                   010E0008  00060 P.AAI:  .LONG    17694728
                                   00000000' 00064         .ADDRESS P.AAJ
                                             00068 P.AAL:  .BLKB    0
                                   010E0000  00068 P.AAK:  .LONG    17694720
                                   00000000' 0006C         .ADDRESS P.AAL
            00 00 54 55 50 54 55 4F           00070 P.AAN:  .ASCII   \OUTPUT\<0><0>
                                   010E0006  00078 P.AAM:  .LONG    17694726
                                   00000000' 0007C         .ADDRESS P.AAN
            00 00 45 54 45 4C 45 44           00080 P.AAP:  .ASCII   \DELETE\<0><0>
                                   010E0006  00088 P.AAO:  .LONG    17694726
                                   00000000' 0008C         .ADDRESS P.AAP
            00 45 43 41 4C 50 45 52           00090 P.AAR:  .ASCII   \REPLACE\<0>
                                   010E0007  00098 P.AAQ:  .LONG    17694727
                                   00000000' 0009C         .ADDRESS P.AAR
            00 00 4D 45 54 53 59 53           000A0 P.AAT:  .ASCII   \SYSTEM\<0><0>
                                   010E0006  000A8 P.AAS:  .LONG    17694726
                                   00000000' 000AC         .ADDRESS P.AAT
```

```
                           00 54 43 45 54 4F 52 50  000B0 P.AAV:  .ASCII  \PROTECT\<0>
                                        010E0007  000B8 P.AAU:  .LONG   17694727
                                        00000000' 000BC          .ADDRESS P.AAV
         00 4E 43 4F 4C 42 5F 54 52 41 54 53  000C0 P.AAX:  .ASCII  \START_BLOCK\<0>
                                        010E000B  000CC P.AAW:  .LONG   17694731
                                        00000000' 000D0          .ADDRESS P.AAX
                           00 00 00 6B 63 6F 6C 62  000D4 P.AAZ:  .ASCII  \block\<0><0><0>
                                        010E0005  000DC P.AAY:  .LONG   17694725
                                        00000000' 000E0          .ADDRESS P.AAZ
                           00 00 64 72 6F 63 65 72  000E4 P.ABB:  .ASCII  \record\<0><0>
                                        010E0006  000EC P.ABA:  .LONG   17694726
                                        00000000' 000F0          .ADDRESS P.ABB

                                          ASCID_ALLOCATION=   P.AAA
                                          ASCID_BEST_TRY=     P.AAC
                                          ASCID_CONTIGUOUS=   P.AAE
                                          ASCID_EXTENSION=    P.AAG
                                          ASCID_TRUNCATE=     P.AAI
                                                      .EXTRN  EXCHSCMD_CLI_GET_INTEGER
                                                      .EXTRN  EXCHSCMD_PARSE_FILESPEC
                                                      .EXTRN  EXCHSDOS11_CREATE_FILE
                                                      .EXTRN  EXCHSDOS11_OPEN_FILE
                                                      .EXTRN  EXCHSFIL11_CREATE_FILE
                                                      .EXTRN  EXCHSFIL11_OPEN_FILE
                                                      .EXTRN  EXCHSMOUN_IMPLIED_MOUNT
                                                      .EXTRN  EXCHSRT11_CREATE_FILE
                                                      .EXTRN  EXCHSRT11_OPEN_FILE
                                                      .EXTRN  EXCHSRT11_WRITE_CLEANUP
                                                      .EXTRN  EXCHSRT11_WRITE_PREPARE
                                                      .EXTRN  EXCHSUTIL_DOS11CTX_RELEASE
                                                      .EXTRN  EXCHSUTIL_FAO_BUFFER
                                                      .EXTRN  EXCHSUTIL_FILB_ALLOCATE
                                                      .EXTRN  EXCHSUTIL_FILB_RELEASE
                                                      .EXTRN  EXCHSUTIL_FILE_ERROR
                                                      .EXTRN  EXCHSUTIL_NAMB_RELEASE
                                                      .EXTRN  EXCHSUTIL_RMSB_ALLOCATE
                                                      .EXTRN  EXCHSUTIL_RMSB_RELEASE
                                                      .EXTRN  EXCHSUTIL_RT11CTX_ALLOCATE
                                                      .EXTRN  EXCHSUTIL_RT11CTX_RELEASE
                                                      .EXTRN  EXCHSUTIL_VM_ALLOCATE
                                                      .EXTRN  EXCHSA_GBL, STRSCOPY_DX
                                                      .EXTRN  EXCHS_PARSEERR, CLISPRESENT
                                                      .EXTRN  CLIS_PRESENT, CLIS_NEGATED
                                                      .EXTRN  EXCHSUTIL_BLOCK_CHECK
                                                      .EXTRN  EXCHS_NOCOPLOCK
                                                      .EXTRN  EXCHS_BADFILENAME
                                                      .EXTRN  EXCHS_MANY_TO_ONE
                                                      .EXTRN  EXCHS_STRTNOMULTI
                                                      .EXTRN  EXCHS_CANCELED, EXCHS_NOTCOPIED
                                                      .EXTRN  EXCHS_PARTCOPIED
                                                      .EXTRN  EXCHS_COPNEWNAME
                                                      .EXTRN  EXCHS_COPIED, EXCHS_NOTCOP_RETRY
                                                      .EXTRN  EXCHS_RT11_OVERFLOW
                                                      .EXTRN  EXCHS_VOLUME_FULL
                                                      .EXTRN  EXCHS_NOCOPSAMDEV
                                                      .EXTRN  EXCHS_ILLMTCOPY
```

```
                                                          .PSECT  EXCHSCOPY_CODE,NOWRT,2

                                     0FFC 00000           .ENTRY  EXCHSCOPY_COPY, Save R2,R3,R4,R5,R6,R7,R8,- ; 0211
                                                                  R9,R10,R1T
                      5E           18 C2 00002           SUBL2   #24, SP
             0000V    CF           00 FB 00005           CALLS   #0, COPY_INIT                                   0260
                      50 00000000G EF D0 0000A           MOVL    EXCH$A_GBL, R0                                  0264
                      55        04 A0 D0 00011           MOVL    4(R0),_COPY
                      0000'        CF 9F 00015           PUSHAB  P.AAK                                           0268
                                1C A5 9F 00019           PUSHAB  28(COPY)
             00000000G 00        02 FB 0001C           CALLS   #2, STRSCOPY_DX
                                AE 9F 00023           PUSHAB  OUT_NAMB                                          0274
                                14 A5 9F 00026           PUSHAB  20(COPY)
                                7E D4 00029           CLRL    -(SP)
                                1C A5 9F 0002B           PUSHAB  28(COPY)                                        0273
                      0000'        CF 9F 0002E           PUSHAB  P.AAM
             00000000G EF        05 FB 00032           CALLS   #5, EXCHSCMD_PARSE_FILESPEC                      0274
                      58        50 D0 00039           MOVL    R0, STATUS
                      1B        58 E8 0003C           BLBS    STATUS, 1$
                      52 00000000G 8F D0 0003F           MOVL    #EXCHS_PARSEERR, TEMP                            0276
                      58 DD 00046           PUSHL   STATUS
                                14 A5 9F 00048           PUSHAB  20(COPY)
                                01 DD 0004B           PUSHL   #1
                                52 DD 0004D           PUSHL   TEMP
             00000000G 00        04 FB 0004F           CALLS   #4, LIBSSIGNAL
                      50        52 D0 00056           MOVL    TEMP, R0
                                04 00059           RET
                      54     14 AE D0 0005A 1$:        MOVL    OUT_NAMB, R4                                     0278
                   48 A5     54 D0 0005E           MOVL    R4,_72(COPY)
                   59     30 A5 9E 00062           MOVAB   48(COPY), R9                                      0282
                      0000'        CF 9F 00066           PUSHAB  ASCID_BEST_TRY
             00000000G 00        01 FB 0006A           CALLS   #1, CCISPRESENT
   69           01        50 F0 00071           INSV    R0, #0, #1, (R9)
                      0000'        CF 9F 00076           PUSHAB  ASCID_CONTIGUOUS                                0283
             00000000G 00        01 FB 0007A           CALLS   #1, CCISPRESENT
   69           01        50 F0 00081           INSV    R0, #1, #1, (R9)
                      0000'        CF 9F 00086           PUSHAB  P.AAO                                           0284
             00000000G 00        01 FB 0008A           CALLS   #1, CLISPRESENT
   69           01        50 F0 00091           INSV    R0, #2, #1, (R9)
                      0000'        CF 9F 00096           PUSHAB  P.AAQ                                           0285
             00000000G 00        01 FB 0009A           CALLS   #1, CLISPRESENT
   69           01        50 F0 000A1           INSV    R0, #7, #1, (R9)
                      0000'        CF 9F 000A6           PUSHAB  P.AAS                                           0286
             00000000G 00        01 FB 000AA           CALLS   #1, CLISPRESENT
   69           01        50 F0 000B1           INSV    R0, #9, #1, (R9)
                      0000'        CF 9F 000B6           PUSHAB  ASCID_TRUNCATE                                  0287
             00000000G 00        01 FB 000BA           CALLS   #1, CCISPRESENT
   69           01        50 F0 000C1           INSV    R0, #10, #1, (R9)
                      0000'        CF 9F 000C6           PUSHAB  P.AAU                                           0291
             00000000G 00        01 FB 000CA           CALLS   #1, CLISPRESENT
   69           01        50 F0 000D1           INSV    PROTECT, #5, #1, (R9)                                  0292
                      52 D4 000D6           CLRL    R2                                                        0293
             00000000G 8F        50 D1 000D8           CMPL    PROTECT, #CLIS_PRESENT
                                02 12 000DF           BNEQ    2$
                      52 D6 000E1           INCL    R2
                      51 D4 000E3 2$:        CLRL    R1                                                        0294
             00000000G 8F        50 D1 000E5           CMPL    PROTECT, #CLIS_NEGATED
```

```
EXCH$COPY        copy verb dispatch and misc routines         G 1
V04-000          exch$copy_copy                  16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742        Page 16
                                                  5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1             (4)
```

```
                                       02  12 000EC              BNEQ    3$
                                       51  D6 000EE              INCL    R1
                   53             51   52  89 000F0  3$:         BISB3   R2, R1, R3
       69          01             06   53  F0 000F4              INSV    R3, #6, #1, (R9)
                                   24  A5  9F 000F9              PUSHAB  36(COPY)
                                 0000' CF  9F 000FC              PUSHAB  ASCID_ALLOCATION
                   00000000G EF        02  FB 00100              CALLS   #2, EXCH$CMD_CLI_GET_INTEGER
                                   58  50  D0 00107              MOVL    R0, STATUS
                                   51  58  E9 0010A              BLBC    STATUS, 6$
                                   28  A5  9F 0010D              PUSHAB  40(COPY)
                                 0000' CF  9F 00110              PUSHAB  ASCID_EXTENSION
                   00000000G EF        02  FB 00114              CALLS   #2, EXCH$CMD_CLI_GET_INTEGER
                                   58  50  D0 0011B              MOVL    R0, STATUS
                                   3D  58  E9 0011E              BLBC    STATUS, 6$
                                   2C  A5  9F 00121              PUSHAB  44(COPY)
                                 0000' CF  9F 00124              PUSHAB  P.AAW
                   00000000G EF        02  FB 00128              CALLS   #2, EXCH$CMD_CLI_GET_INTEGER
                                   58  50  D0 0012F              MOVL    R0, STATUS
                                   29  58  E9 00132              BLBC    STATUS, 6$
                               74  A4  D5 00135              TSTL    116(R4)
                                   30  12 00138              BNEQ    7$
                   05          6B  A4  E8 0013A              BLBS    107(R4), 4$
       27          6A  A4      03  E0 0013E              BBS     #3, 106(R4), 7$
                   01      78  A4  91 00143  4$:         CMPB    120(R4), #1
                                   06  13 00147              BEQL    5$
                   02      78  A4  91 00149              CMPB    120(R4), #2
                                   1B  12 0014D              BNEQ    7$
                                   54  DD 0014F  5$:         PUSHL   R4
                   00000000G EF        01  FB 00151              CALLS   #1, EXCH$MOUN_IMPLIED_MOUNT
                                   58  50  D0 00158              MOVL    R0, STATUS
                                   0C  58  E8 0015B              BLBS    STATUS, 7$
                                   54  DD 0015E  6$:         PUSHL   R4
                   00000000G EF        01  FB 00160              CALLS   #1, EXCH$UTIL_NAMB_RELEASE
                               0334  31 00167              BRW     55$
                               74  A4  D5 0016A  7$:         TSTL    116(R4)
                                   65  13 0016D              BEQL    11$
                   53      74  A4  D0 0016F              MOVL    116(R4), R3
                   52 041B00F3  8F  D0 00173              MOVL    #68878579, R2
                   51      01F0  8F  3C 0017A              MOVZWL  #496, R1
                                   53  D0 0017F              MOVL    R3, R0
                   00000000G  EF  16 00182              JSB     EXCH$UTIL_BLOCK_CHECK
       26          48  A3      05  E0 00188              BBS     #5, 72(R3), 8$
                               69  A3  9F 0018D              PUSHAB  105(R3)
                               65  A3  DD 00190              PUSHL   101(R3)
                                   02  DD 00193              PUSHL   #2
                   00000000G  8F  DD 00195              PUSHL   #EXCH$_NOCOPLOCK
                   00000000G 00        04  FB 0019B              CALLS   #4, LIB$SIGNAL
                                   54  DD 001A2              PUSHL   R4
                   00000000G EF        01  FB 001A4              CALLS   #1, EXCH$UTIL_NAMB_RELEASE
                   50 00000000G  8F  D0 001AB              MOVL    #EXCH$_NOCOPLOCK, R0
                                   04 001B2              RET
                   03          00  58  A3  8F 001B3  8$:         CASEB   88(R3), #0, #3
       000A        0050        001E    0050 001B8  9$:         .WORD   14$-9$,-
                                                                       12$-9$,-
                                                                       14$-9$,-
                                                                       10$-9$
                               46  11 001C0              BRB     14$
```

```
0300
0307
0314
0323
0328
0331
0333
0337
0340
0341
0347
0355
0359
0362
0363
0364
0367
```

```
                    19    6E  A4 E8 001C2  10$:   BLBS    110(R4), 13$                      0372
              14    6E  A4  02 E0 001C6           BBS     #2, 110(R4), 13$                  0374
                            53 DD 001CB           PUSHL   R3                                0382
         00000000G EF       01 FB 001CD           CALLS   #1, EXCH$RT11_WRITE_PREPARE
                            32 11 001D4  11$:      BRB     14$                               0367
              29    05    6E  A4 E8 001D6  12$:    BLBS    110(R4), 13$                      0387
              29    6E  A4  01 E1 001DA           BBC     #1, 110(R4), 14$                   0389
                      5D A3 9F 001DF  13$:         PUSHAB  93(R3)                            0393
                      59 A3 DD 001E2              PUSHL   89(R3)
                      10 A4 9F 001E5              PUSHAB  16(R4)
                         03 DD 001E8              PUSHL   #3
      00000000G 8F DD 001EA              PUSHL   #EXCH$_BADFILENAME
         00000000G 00  05 FB 001F0              CALLS   #5, LIB$SIGNAL
                      54 DD 001F7              PUSHL   R4
         00000000G EF  01 FB 001F9              CALLS   #1, EXCH$UTIL_NAMB_RELEASE           0394
                   50 00000000G 8F D0 00200              MOVL    #EXCH$_BADFILENAME, R0      0395
                         04 00207              RET
         00000000G EF  00 FB 00208  14$:      CALLS   #0, EXCH$UTIL_FILB_ALLOCATE            0406
                      53 50 D0 0020F              MOVL    R0, OUT_FILB
              44 A5  53 D0 00212              MOVL    OUT_FILB, 68(COPY)                     0407
                      53 DD 00216              PUSHL   OUT_FILB                              0408
                      54 DD 00218              PUSHL   R4
         0000V CF  02 FB 0021A              CALLS   #2, EXCH$COPY_NAMB_TO_FILB
                   0C AE D4 0021F              CLRL    ABORT                                 0414
                      58 01 D0 00222              MOVL    #1, STATUS                         0415
                   6E 34 A5 9E 00225              MOVAB   52(COPY), (SP)                     0426
         0000V CF  00 FB 00229  15$:      CALLS   #0, COPY_PARSE_NEXT_INPUT                  0416
                   10 AE 50 D0 0022E              MOVL    R0, PRS_STAT
                   03 10 AE E8 00232              BLBS    PRS_STAT, 16$
                      0257 31 00236              BRW     53$
              52 3C A5 D0 00239  16$:      MOVL    60(COPY), INP_FILB                        0422
              34 00 BE E9 0023D              BLBC    20(SP), 19$                             0426
              13 6C A4 E8 00241              BLBS    108(R4), 17$                            0432
         0E 6D A4  01 E1 00245              BBC     #1, 109(R4), 17$                         0434
         09 6D A4  02 E1 0024A              BBC     #2, 109(R4), 17$                         0436
              58 00000000G 8F D0 0024F              MOVL    #EXCH$_MANY_TO_ONE, STATUS       0440
                      0C 11 00256              BRB     18$                                   0441
                   2C A5 D5 00258  17$:      TSTL    44(COPY)                                0448
                      18 13 0025B              BEQL    19$
              58 00000000G 8F D0 0025B              MOVL    #EXCH$_STRTNOMULTI, STATUS       0451
                      58 DD 00264  18$:      PUSHL   STATUS                                  0452
         00000000G 00  01 FB 00266              CALLS   #1, LIB$SIGNAL
         0000V CF  00 FB 0026D              CALLS   #0, COPY_PARSE_CLEANUP                   0453
                      0217 31 00272              BRW     52$                                 0450
                   1C 00000000G FF E9 00275  19$:  BLBC    @EXCH$A_GBL, 20$                  0463
              08 AE 00000000G 8F D0 0027C              MOVL    #EXCH$_CANCELED, INO_STAT     0466
              50 08 AE D0 00284              MOVL    INO_STAT, STATUS2                       0467
         50 03 00 03 F0 00288              INSV    #3, #0, #3, STATUS2
                   50 DD 0028D              PUSHL   STATUS2
         00000000G 00  01 FB 0028F              CALLS   #1, LIB$SIGNAL
                      09 11 00296              BRB     21$
         0000V CF  00 FB 00298  20$:      CALLS   #0, COPY_INPUT_OPEN                        0463
              08 AE 50 D0 0029D              MOVL    R0, INO_STAT                            0470
   00 50 00 BE  02 EF 002A1  21$:      EXTZV   #2, #1, 20(SP), R0                            0474
00 BE 01  03 F0 002A7              INSV    R0, #3, #1, 20(SP)
                   00 BE 04 8A 002AD              BICB2   #4, 20(SP)                         0475
                      03 08 AE E8 002B1              BLBS    INO_STAT, 22$                    0477
```

```
EXCH$COPY      copy verb dispatch and misc routines          16-Sep-1984 00:41:48    VAX-11 Bliss-32 V4.0-742      Page 18
V04-000        exch$copy_copy                                  5-Sep-1984 22:04:55    [EXCHNG.SRC]EXCCOPY.B32;1           (4)
```

```
                                01A4  31 002B5           BRW     50$
                    0000V  CF   00  FB 002B8  22$:       CALLS   #0, COPY_OUTPUT_CREATE              0486
                           56   50  D0 002BD             MOVL    R0, CRE_STAT
                           03   56  E8 002C0             BLBS    CRE_STAT, 23$
                                0137 31 002C3            BRW     46$
                           5A   D4 002C6  23$:           CLRL    GETPUT_ERR                         0497
                           57   D4 002C8                 CLRL    PUT_STAT
                      04   AE   D4 002CA                 CLRL    REC_COUNT
                           52   DD 002CD  24$:           PUSHL   INP_FILB                           0498
                    52  B2  01  FB 002CF                 CALLS   #1, @82(INP_FILB)
                           5B   D0 002D3                 MOVL    R0, GET_STAT
                           5B   E9 002D6                 BLBC    GET_STAT, 26$
                    56  B3  00  FB 002D9                 CALLS   #0, @86(OUT_FILB)                  0501
                           57   D0 002DD                 MOVL    R0, PUT_STAT
                           57   E9 002E0                 BLBC    PUT_STAT, 25$
                      04   AE  D6 002E3                   INCL    REC_COUNT                          0502
            E0 00000000G  FF  E9 002E6                   BLBC    @EXCH$A_GBL, 24$                   0506
            57 00000000G  8F  D0 002ED                   MOVL    #EXCH$_CANCELED, PUT_STAT          0509
                      0C   AE  01  D0 002F4              MOVL    #1, ABORT                          0510
                           57   D0 002F8                 MOVL    PUT_STAT, STATUS2                  0511
   50       03             00   03 FO 002FB             INSV    #3, #0, #3, STATUS2
                           50   DD 00300                 PUSHL   STATUS2
            00000000G  00  01  FB 00302                  CALLS   #1, LIB$SIGNAL
                           5B   E8 00309  25$:           BLBS    GET_STAT, 27$                     0518
                      0A   5B  D5 0030C  26$:            TSTL    GET_STAT
                      06   13 0030E                      BEQL    27$
                      58   5B  D0 00310                  MOVL    GET_STAT, STATUS                   0521
                      5A   01  D0 00313                  MOVL    #1, GETPUT_ERR                     0522
                      0A   57  E8 00316  27$:            BLBS    PUT_STAT, 28$                      0525
                           57   D5 00319                 TSTL    PUT_STAT
                      06   13 0031B                      BEQL    28$
                      58   57  D0 0031D                  MOVL    PUT_STAT, STATUS                   0528
                      5A   01  D0 00320                  MOVL    #1, GETPUT_ERR                     0529
         31   2B  A3  02  E0 00323  28$:                 BBS     #2, 43(OUT_FILB), 32$             0534
                      1B   5A  E9 00328                  BLBC    GETPUT_ERR, 30$                    0537
                      04   AE  D5 0032B                  TSTL    REC_COUNT                         0539
                      0C   13 0032E                      BEQL    29$
         07   2B  A3  06  E0 00330                       BBS     #6, 43(OUT_FILB), 29$             0541
            0A 00000000G  FF  E9 00335                   BLBC    @EXCH$A_GBL, 30$                  0543
                    0000V  CF  00  FB 0033C  29$:        CALLS   #0, COPY_OUTPUT_DELETE            0546
                      04   AE  D4 00341                   CLRL    REC_COUNT                         0547
                      0E   11 00344                       BRB     31$                              0537
                    0000V  CF  00  FB 00346  30$:        CALLS   #0, COPY_OUTPUT_CLOSE            0556
                      06   50  E8 0034B                  BLBS    CLS_STAT, 31$                     0558
                      58   50  D0 0034E                  MOVL    CLS_STAT, STATUS                  0561
                      5A   01  D0 00351                  MOVL    #1, GETPUT_ERR                    0562
         07   2B  A3  02  E1 00354  31$:                 BBC     #2, 43(OUT_FILB), 33$            0570
                      04   AE  D4 00359  32$:            CLRL    REC_COUNT                         0573
         2B  A3  04  8A 0035C                            BICB2   #4, 43(OUT_FILB)                 0574
                      50   D4 00360  33$:                CLRL    COP_STAT                         0579
                      5A   E9 00362                      BLBC    GETPUT_ERR, 35$                   0580
                      04   AE  D5 00365                  TSTL    REC_COUNT                         0583
                      09   12 00368                      BNEQ    34$
         50 00000000G  8F  D0 0036A                      MOVL    #EXCH$_NOTCOPIED, COP_STAT       0585
                      2B   11 00371                      BRB     38$
         50 00000000G  8F  D0 00373  34$:                MOVL    #EXCH$_PARTCOPIED, COP_STAT      0587
                      22   11 0037A                      BRB     38$                              0580
```

```
EXCHSCOPY          copy verb dispatch and misc routines         J 1          VAX-11 Bliss-32 V4.0-742        Page 19
V04-000            exch$copy_copy                      16-Sep-1984 00:41:48                                    (4)
                                                        5-Sep-1984 22:04:55    [EXCHNG.SRC]EXCCOPY.B32;1
```

```
                              2B   A3 95 0037C 35$:   TSTB    43(OUT_FILB)                          : 0591
                                   0D 18 0037F         BGEQ    36$
            09              69     04 E0 00381         BBS     #4, (R9), 36$                          : 0593
                            50 00000000G 8F DO 00385   MOVL    #EXCH$_COPNEWNAME, COP_STAT           : 0595
                                   10 11 0038C         BRB     38$
            05              69     03 E0 0038E 36$:    BBS     #3, (R9), 37$                          : 0596
            07       00     BE     03 E1 00392         BBC     #3, a0(SP), 38$                        : 0598
                            50 00000000G 8F DO 00397 37$: MOVL  #EXCH$_COPIED, COP_STAT              : 0600
            07       00     BE     02 E1 0039E 38$:    BBC     #2, a0(SP), 39$                        : 0605
                            50 00000000G 8F DO 003A3   MOVL    #EXCH$_NOTCOP_RETRY, COP_STAT          : 0607
                            02 00000000G FF E9 003AA 39$: BLBC  aEXCH$A_GBL, 40$                     : 0611
                                   50 D4 003B1         CLRL    COP_STAT                              : 0613
                                   50 D5 003B3 40$:    TSTL    COP_STAT                              : 0617
                                   44 13 003B5         BEQL    45$
                     02     28     A3 91 003B7         CMPB    40(OUT_FILB), #2                       : 0622
                            0A     12 003BB            BNEQ    41$
            00000200       8F     35 A3 D1 003BD       CMPL    53(OUT_FILB), #512                     : 0623
                            0C     13 003C5            BEQL    42$
                     01     29     A3 91 003C7 41$:    CMPB    41(OUT_FILB), #1                       : 0625
                            06     13 003CB            BEQL    42$
                     01     29     A2 91 003CD         CMPB    41(INP_FILB), #1                       : 0626
                            07     12 003D1            BNEQ    43$
                     51     0000'  CF 9E 003D3 42$:    MOVAB   P.AAY, B_OR_R                          : 0627
                            05     11 003D8            BRB     44$
                     51     0000'  CF 9E 003DA 43$:    MOVAB   P.ABA, B_OR_R
                                   51 DD 003DF 44$:    PUSHL   B_OR_R                                 : 0631
                            08     AE DD 003E1         PUSHL   REC_COUNT
                            5A     A3 9F 003E4         PUSHAB  90(OUT_FILB)
                            3A     A3 DD 003E7         PUSHL   58(OUT_FILB)
                            5A     A2 9F 003EA         PUSHAB  90(INP_FILB)
                            3A     A2 DD 003ED         PUSHL   58(INP_FILB)
                            06     DD 003F0            PUSHL   #6
                                   50 DD 003F2         PUSHL   COP_STAT
            00000000G 00   08 FB 003F4               CALLS   #8, LIB$SIGNAL
                            53     11 003FB 45$:       BRB     48$                                   : 0486
                            56     DD 003FD 46$:       PUSHL   CRE_STAT                              : 0640
                            5A     A3 9F 003FF         PUSHAB  90(OUT_FILB)
                            3A     A3 DD 00402         PUSHL   58(OUT_FILB)
                            5A     A2 9F 00405         PUSHAB  90(INP_FILB)
                            3A     A2 DD 00408         PUSHL   58(INP_FILB)
                            04     DD 0040B            PUSHL   #4
                 00000000G 8F DD 0040D               PUSHL   #EXCH$_NOTCOPIED
            00000000G 00   07 FB 00413               CALLS   #7, LIB$SIGNAL
                            56     DO 0041A            MOVL    CRE_STAT, STATUS                       : 0642
                            2D     13 0041D            BEQL    47$                                   : 0649
            000184C4       8F     56 D1 0041F         CMPL    CRE_STAT, #99524
                            24     13 00426            BEQL    47$
            00000000G       8F     56 D1 00428         CMPL    CRE_STAT, #EXCH$_RT11_OVERFLOW
                            1B     13 0042F            BEQL    47$
            00000000G       8F     56 D1 00431         CMPL    CRE_STAT, #EXCH$_VOLUME_FULL
                            12     13 00438            BEQL    47$
            00000000G       8F     56 D1 0043A         CMPL    CRE_STAT, #EXCH$_NOCOPSAMDEV
                            09     13 00441            BEQL    47$
            00000000G       8F     56 D1 00443         CMPL    CRE_STAT, #EXCH$_ILLMTCOPY
                            04     12 0044A            BNEQ    48$
                     0C     AE     01 DO 0044C 47$:    MOVL    #1, ABORT                             : 0651
                     0000V  CF     00 FB 00450 48$:    CALLS   #0, COPY_INPUT_CLOSE                  : 0658
```

; R

```
EXCH$COPY      copy verb dispatch and misc routines          16-Sep-1984 00:41:48    VAX-11 Bliss-32 V4.0-742        Page 20
V04-000        exch$copy_copy                                 5-Sep-1984 22:04:55     [EXCHNG.SRC]EXCCOPY.B32;1             (4)
```

```
                        27      OC   AE  E8 00455         BLBS    ABORT, 51$                      ; 0659
                              FE19   31 00459 49$:        BRW     19$                             ; 
                        08      AE   D5 0045C 50$:        TSTL    INO_STAT                        ; 0668
                              1F     13 0045F             BEQL    51$
                        18 00000000G FF E8 00461          BLBS    @EXCH$A_GBL, 51$                ; 0670
                        58      08   AE  D0 00468          MOVL    INO_STAT, STATUS               ; 0675
             00018292   8F      08   AE  D1 0046C          CMPL    INO_STAT, #98962              ; 0678
                        0A     13 00474                    BEQL    51$
             000184C4   8F      08   AE  D1 00476          CMPL    INO_STAT, #99524
                              D9     12 0047E              BNEQ    49$
             0000V      CF     00   FB 00480 51$:          CALLS   #0, COPY_PARSE_CLEANUP        ; 0688
                        03      OC   AE  E8 00485          BLBS    ABORT, 52$                     ; 0689
                              FD9D   31 00489              BRW     15$
                        09      10   AE  E8 0048C 52$:     BLBS    PRS_STAT, 54$                  ; 0695
                        10      AE   D5 00490 53$:         TSTL    PRS_STAT
                        04     13 00493                    BEQL    54$
                        58      10   AE  D0 00495          MOVL    PRS_STAT, STATUS
             0000V      CF     00   FB 00499 54$:          CALLS   #0, COPY_OUTPUT_CLEANUP       ; 0699
                        50      58   D0 0049E 55$:         MOVL    STATUS, R0                     ; 0702
                              04 004A1                     RET                                    ; 0703

; Routine Size:  1186 bytes,     Routine Base:  EXCH$COPY_CODE + 0000
```

```
EXCH$COPY          copy verb dispatch and misc routines           16-Sep-1984 00:41:48    VAX-11 Bliss-32 V4.0-742           Page 21
V04-000            exch$copy_init                                 5-Sep-1984 22:04:55     [EXCHNG.SRC]EXCCOPY.B32;1                  (5)
```

```
612    0704   1 GLOBAL ROUTINE copy_init : NOVALUE =     %SBTTL 'exch$copy_init'
613    0705   2 BEGIN
614    0706   2 !++
615    0707   2 !
616    0708   2 ! FUNCTIONAL DESCRIPTION:
617    0709   2 !
618    0710   2 !     Common init routine for the copy and type verbs
619    0711   2 !
620    0712   2 ! INPUTS:
621    0713   2 !
622    0714   2 !     none
623    0715   2 !
624    0716   2 ! IMPLICIT INPUTS:
625    0717   2 !
626    0718   2 !     Command parameters and qualifiers as returned from CLI$ routines.  Global environment ref'd by exch$
627    0719   2 !
628    0720   2 ! OUTPUTS:
629    0721   2 !
630    0722   2 !     none
631    0723   2 !
632    0724   2 ! IMPLICIT OUTPUTS:
633    0725   2 !
634    0726   2 !     none
635    0727   2 !
636    0728   2 ! ROUTINE VALUE:
637    0729   2 !
638    0730   2 !     none
639    0731   2 !
640    0732   2 ! SIDE EFFECTS:
641    0733   2 !
642    0734   2 !     Files may be created.
643    0735   2 !--
644    0736   2
645    0737   2 $dbgtrc_prefix ('copy_init> ');
646    0738   2
647    0739   2 LOCAL
648    0740   2     status
649    0741   2     ;
650    0742   2
651    0743   2 BIND
652    0744   2     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock  ! Pointer to work area
653    0745   2     ;
654    0746   2
655    0747   2
656    0748   2 ! If our pointer is null, we need to allocate and initialize the work area
657    0749   2 !
658    0750   2 IF .copy EQL 0
659    0751   2 THEN
660    0752   3     BEGIN
661    0753   3
662    0754   3     ! Get the right sized chunk of memory
663    0755   3     !
664    0756   3     copy = exch$util_vm_allocate (exchblk$s_copy);
665    0757   3     ! Set the ident fields
666    0758   3     !
667    0759   3     $block_init (.copy, copy);
668    0760   3
```

```
669   0761   3             ! Set the dynamic strings
670   0762   3             !
671   0763   3
672   0764   3             $dyn_str_desc_init (copy [copy$q_default_filename]);
673   0765   3             $dyn_str_desc_init (copy [copy$q_input_filename]);
674   0766   3             $dyn_str_desc_init (copy [copy$q_output_filename]);
675   0767   3             $dyn_str_desc_init (copy [copy$q_input_sticky_name]);
676   0768   3     !\      $dyn_str_desc_init (copy [copy$q_q_boot]);
677   0769   3     !\      $dyn_str_desc_init (copy [copy$q_q_fdl]);
678   0770   3     !\      $dyn_str_desc_init (copy [copy$q_q_protection]);
679   0771   3
680   0772   3             END
681   0775   2         ELSE
682   0774   2             BEGIN
683   0775   3
684   0776   3             ! Free the dynamic strings and the Chicago 7
685   0777   3             !
686   0778   3             str$free1_dx (copy [copy$q_default_filename]);
687   0779   3             str$free1_dx (copy [copy$q_input_filename]);
688   0780   3             str$free1_dx (copy [copy$q_output_filename]);
689   0781   3     !\      str$free1_dx (copy [copy$q_q_boot]);
690   0782   3     !\      str$free1_dx (copy [copy$q_q_fdl]);
691   0783   3     !\      str$free1_dx (copy [copy$q_q_protection]);
692   0784   3
693   0785   2             END;
694   0786   2
695   0787   2         ! Get some confidence that our work area is valid
696   0788   2
697   0789   2         $block_check (2, .copy, copy, 408);
698   0790   2
699   0791   2         ! Set the last part of the block to nulls
700   0792   2         !
701   0793   2         CH$FILL (0, copy$k_end_zero - copy$k_start_zero, .copy + copy$k_start_zero);
702   0794   2
703   0795   2         ! Start with a very large max rec, it will be adjusted if too large
704   0796   2         !
705   0797   2         copy [copy$l_max_rec] = 65535;
706   0798   2
707   0799   2         ! Get the global boolean qualifiers common to both commands
708   0800   2
709   0801   2         status = cli$present (%ASCID 'LOG');                           ! Global qualifier
710   0802   2         copy [copy$v_q_log]        = .status;                          ! Log state
711   0803   2         copy [copy$v_q_nolog_explicit] = (.status EQL cli$_negated);   ! Set if /NOLOG is present
712   0804   2
713   0805   2     !\  copy [copy$v_q_confirm]        = cli$present (%ASCID 'CONFIRM');                    ! global
714   0806   2
715   0807   2         RETURN;
716   0808   1         END;
```

```
                                         .PSECT   EXCH$COPY_PLIT,NOWRT,2

                    00 47 4F  4C  000F4  P.ABD:   .ASCII   \LOG\<0>
                       010E0003  000F8  P.ABC:   .LONG    17694723
                       00000000' 000FC           .ADDRESS P.ABD
```

EXCH$COPY   copy verb dispatch and misc routines   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 23
V04-000   exch$copy_init   5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (5)

N 1

```
                                                              .EXTRN    EXCH$GQ_DYN_STR_TEMPLATE
                                                              .EXTRN    STR$FREE1_DX

                                                              .PSECT    EXCH$COPY_CODE,NOWRT,2

                                        OOFC  00000           .ENTRY    COPY_INIT, Save R2,R3,R4,R5,R6,R7    ; 0704
                          57 00000000G  00  9E  00002         MOVAB     STR$FREE1_DX, R7
             52 00000000G EF             04  C1  00009         ADDL3     #4, EXCH$A_GBL, R2                   ; 0744
                                        62  D5  00011         TSTL      (R2)                                ; 0750
                                        52  12  00013         BNEQ      1$
                          7E        4C  8F  9A  00015         MOVZCL    #76, -(SP)                          ; 0756
             00000000G EF               01  FB  00019         CALLS     #1, EXCH$UTIL_VM_ALLOCATE
                                        62          50  D0  00020  MOVL  R0, (R2)
                       08 A0        4C  8F  9B  00023         MOVZBW    #76, 8(R0)                          ; 0760
                       0A A0            01  8E  00028         MNEGB     #1, 10(R0)
                          50            62  D0  0002C         MOVL      (R2), R0                            ; 0764
                          53 00000000G EF  D0  0002F         MOVL      TMPL, R3
                          60            53  D0  00036         MOVL      R3, (R0)
                          51 00000000G EF  D0  00039         MOVL      TMPL+4, R1
                       04 A0            51  D0  00040         MOVL      R1, 4(R0)
             50                        0C  C1  00044         ADDL3     #12, (R2), R0                       ; 0765
                          60            53  D0  00048         MOVL      R3, (R0)
                       04 A0            51  D0  0004B         MOVL      R1, 4(R0)
             50                        14  C1  0004F         ADDL3     #20, (R2), R0                       ; 0766
                          60            53  D0  00053         MOVL      R3, (R0)
                       04 A0            51  D0  00056         MOVL      R1, 4(R0)
             50                        1C  C1  0005A         ADDL3     #28, (R2), R0                       ; 0767
                          60            53  D0  0005E         MOVL      R3, (R0)
                       04 A0            51  D0  00061         MOVL      R1, 4(R0)
                                        13  11  00065         BRB       2$                                 ; 0750
                                        62  DD  00067  1$:    PUSHL     (R2)                                ; 0778
                       67               01  FB  00069         CALLS     #1, STR$FREE1_DX                    ; 0779
             7E                        0C  C1  0006C         ADDL3     #12, (R2), -(SP)
                       67               01  FB  00070         CALLS     #1, STR$FREE1_DX                    ; 0780
             7E                        14  C1  00073         ADDL3     #20, (R2), -(SP)
                       67               01  FB  00077         CALLS     #1, STR$FREE1_DX
                          56            62  D0  0007A  2$:    MOVL      (R2), R6                            ; 0789
                          52 004C00FF  8F  D0  0007D         MOVL      #4980991, R2
                          51     0198  8F  3C  00084         MOVZWL    #408, R1
                          50            56  D0  00089         MOVL      R6, R0
                             00000000G EF  16  0008C         JSB       EXCH$UTIL_BLOCK_CHECK
                 28    00               6E  00  2C  00092     MOVC5     #0, (SP), #0, #40, 36(R6)           ; 0793
                                        24  A6      00097
                    38 A6        FFFF  8F  3C  00099         MOVZWL    #65535, 56(R6)                      ; 0797
                                  0000' CF  9F  0009F         PUSHAB    P.ABC                              ; 0801
             00000000G 00              01  FB  000A3         CALLS     #1, CLI$PRESENT
        30 A6  01              03      50  F0  000AA         INSV      STATUS, #3, #1, 48(R6)              ; 0802
                                        51  D4  000B0         CLRL      R1                                 ; 0803
             00000000G 8F              50  D1  000B2         CMPL      STATUS, #CLI$_NEGATED
                                        02  12  000B9         BNEQ      3$
                                        51  D6  000BB         INCL      R1
        30 A6  01              04      51  F0  000BD  3$:    INSV      R1, #4, #1, 48(R6)
                                        04  000C3         RET                                             ; 0808

; Routine Size:  196 bytes,    Routine Base:  EXCH$COPY_CODE + 04A2
```

```
718    0809  1  GLOBAL ROUTINE copy_input_close : NOVALUE =     %SBTTL 'copy_input_close'
719    0810  2  BEGIN
720    0811     !++
721    0812     !
722    0813  2  ! FUNCTIONAL DESCRIPTION:
723    0814     !
724    0815  2  !     Close the input file
725    0816     !
726    0817  2  ! INPUTS:
727    0818     !
728    0819  2  !     none
729    0820     !
730    0821  2  ! IMPLICIT INPUTS:
731    0822     !
732    0823  2  !     copy [copy$a_inp_filb] describes the file to be closed
733    0824     !
734    0825  2  ! OUTPUTS:
735    0826     !
736    0827  2  !     none
737    0828     !
738    0829  2  ! IMPLICIT OUTPUTS:
739    0830     !
740    0831  2  !     none
741    0832     !
742    0833  2  ! ROUTINE VALUE:
743    0834     !
744    0835  2  !     Success or worst error encountered.
745    0836     !
746    0837  2  ! SIDE EFFECTS:
747    0838     !
748    0839  2  !     none
749    0840  2  !--
750    0841
751    0842  2  $dbgtrc_prefix ('copy_input_close> ');
752    0843
753    0844  2  LOCAL
754    0845  2      status
755    0846  2      ;
756    0847
757    0848  2  BIND
758    0849  2      copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
759    0850  2      inp_filb = copy [copy$a_inp_filb]    : $ref_bblock  ! Filb for the input
760    0851  2      ;
761    0852
762    0853
763    0854  2  $block_check (2, .copy, copy, 509);
764    0855  2  $block_check (2, .inp_filb, filb, 510);
765    0856
766    0857  2  ! Call the file-specific close routine
767    0858  2  !
768    0859  2  (.inp_filb [filb$a_close_routine]) (.inp_filb);
769    0860
770    0861  2  RETURN;
771    0862  1  END;
```

```
                                       003C 00000          .ENTRY   COPY_INPUT_CLOSE, Save R2,R3,R4,R5       : 0809
                      55 00000000G  EF  9E 00002          MOVAB    EXCH$UTIL_BLOCK_CHECK, R5
        53 00000000G EF               04  C1 00009          ADDL3    #4, EXCH$A_GBL, R3                        : 0849
        54                            63  3C C1 00011        ADDL3    #60, (R3), R4                            : 0850
                      52 004C00FF  8F  D0 00015          MOVL     #4980991, R2                              : 0854
                      51     01FD  8F  3C 0001C          MOVZWL   #509, R1
                      50               63  D0 00021        MOVL     (R3), R0
                                       65  16 00024        JSB      EXCH$UTIL_BLOCK_CHECK
                      53               64  D0 00026        MOVL     (R4), R3                                 : 0855
                      52 035B00FA  8F  D0 00029          MOVL     #56295674, R2
                      51     01FE  8F  3C 00030          MOVZWL   #510, R1
                      50               53  D0 00035        MOVL     R3, R0
                                       65  16 00038        JSB      EXCH$UTIL_BLOCK_CHECK
                                       53  DD 0003A        PUSHL    R3                                       : 0859
                   4A  B3              01  FB 0003C        CALLS    #1, @74(R3)
                                           04 00040        RET                                              : 0862
```

; Routine Size: 65 bytes,   Routine Base: EXCH$COPY_CODE + 0566

```
773    0863  1 GLOBAL ROUTINE copy_input_open =              %SBTTL 'copy_input_open'
774    0864  2 BEGIN
775    0865  2 !+
776    0866  2 !
777    0867  2 ! FUNCTIONAL DESCRIPTION:
778    0868  2 !
779    0869  2 !     Open the input file
780    0870  2 !
781    0871  2 ! INPUTS:
782    0872  2 !
783    0873  2 !     none
784    0874  2 !
785    0875  2 ! IMPLICIT INPUTS:
786    0876  2 !
787    0877  2 !     copy [copy$a_inp_filb] describes the file to be opened
788    0878  2 !
789    0879  2 ! OUTPUTS:
790    0880  2 !
791    0881  2 !     none
792    0882  2 !
793    0883  2 ! IMPLICIT OUTPUTS:
794    0884  2 !
795    0885  2 !     none
796    0886  2 !
797    0887  2 ! ROUTINE VALUE:
798    0888  2 !
799    0889  2 !     Success or worst error encountered.
800    0890  2 !
801    0891  2 ! SIDE EFFECTS:
802    0892  2 !
803    0893  2 !     none
804    0894  2 !--
805    0895  2
806    0896  2 $dbgtrc_prefix ('copy_input_open> ');
807    0897  2
808    0898  2 LOCAL
809    0899  2     status
810    0900  2     ;
811    0901  2
812    0902  2 BIND
813    0903  2     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
814    0904  2     inp_filb = copy [copy$a_inp_filb]    : $ref_bblock, ! Filb for the input
815    0905  2     inp_namb = copy [copy$a_inp_namb]    : $ref_bblock  ! Namb for the input
816    0906  2     ;
817    0907  2
818    0908  2
819    0909  2 $block_check (2, .copy, copy, 409);
820    0910  2 $block_check (2, .inp_filb, filb, 410);
821    0911  2 $block_check (2, .inp_namb, namb, 411);
```

```
 823    0912   2  ! Perform the volume-specific open operation
 824    0913   2  !
 825    0914   2  CASE .inp_namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
 826    0915   2  SET
 827    0916   2      [volb$k_vfmt_dos11]        : status = exch$dos11_open_file ();
 828    0917   2      [volb$k_vfmt_files11]      : status = exch$fil11_open_file ();
 829    0918   2      [volb$k_vfmt_rt11]         : status = exch$rt11_open_file ();
 830    0919   2  !\  [volb$k_vfmt_rtmt]         : $exch_signal_stop (exch$_notimplement);
 831    0920   2      [OUTRANGE,INRANGE]         : $logic_check (0, (false), 233);
 832    0921   2  TES;
 833    0922   2
 834    0923   2  RETURN .status;
 835    0924   1  END;
```

```
                                                                      .EXTRN   EXCH$_BADLOGIC

                                               007C 00000             .ENTRY   COPY_INPUT_OPEN, Save R2,R3,R4,R5,R6           0863
                          56 00000000G EF   9E 00002                  MOVAB    EXCH$UTIL_BLOCK_CHECK, R6
            53 00000000G EF              04 C1 00009                  ADDL3    #4, EXCH$A_GBL, R3                             0903
            55             63              3C C1 00011                  ADDL3    #60, (R3), R5                                 0904
            54             63    00000040  8F C1 00015                  ADDL3    #64, (R3), R4                                 0905
                          52 004C00FF  8F D0 0001D                  MOVL     #4980991, R2                                   0909
                          51      0199  8F 3C 00024                  MOVZWL   #409, R1
                          50        63     D0 00029                  MOVL     (R3), R0
                          66              16 0002C                  JSB      EXCH$UTIL_BLOCK_CHECK
                          52 035B00FA  8F D0 0002E                  MOVL     #56295674, R2                                  0910
                          51      019A  8F 3C 00035                  MOVZWL   #410, R1
                          50        65     D0 0003A                  MOVL     (R5), R0
                          66              16 0003D                  JSB      EXCH$UTIL_BLOCK_CHECK
                          53        64     D0 0003F                  MOVL     (R4), R3                                       0911
                          52 010A00F7  8F D0 00042                  MOVL     #17432823, R2
                          51      019B  8F 3C 00049                  MOVZWL   #411, R1
                          50        53     D0 0004E                  MOVL     R3, R0
                          66              16 00051                  JSB      EXCH$UTIL_BLOCK_CHECK
                03         00    7A A3  8F 00053                  CASEB    122(R3), #0, #3                                0914
 002C     0024       001C      0008       00058 1$:  .WORD    2$-1$,-
                                                                       3$-1$,-
                                                                       4$-1$,-
                                                                       5$-1$

                          7E      E9  8F 9A 00060 2$:  MOVZBL   #233, -(SP)                                    0920
                                   01     DD 00064                  PUSHL    #1
                      00000000G        8F DD 00066                  PUSHL    #EXCH$_BADLOGIC
            00000000G 00              03 FB 0006C                  CALLS    #3, LIB$STOP
                                   04     00073                  RET
            00000000G EF              00 FB 00074 3$:  CALLS    #0, EXCH$DOS11_OPEN_FILE                       0916
                                   04     0007B                  RET
            00000000G EF              00 FB 0007C 4$:  CALLS    #0, EXCH$FIL11_OPEN_FILE                       0917
                                   04     00083                  RET
            00000000G EF              00 FB 00084 5$:  CALLS    #0, EXCH$RT11_OPEN_FILE                        0918
                                   04     0008B                  RET                                                         0924
```

```
; Routine Size:  140 bytes.    Routine Base:  EXCH$COPY_CODE + 05A7
```

```
  837    0925  1 GLOBAL ROUTINE exch$copy_namb_to_filb (namb : $ref_bblock,        %SBTTL 'exch$copy_namb_to_filb (namb, filb)'
  838    0926  1                                        filb : $ref_bblock) : NOVALUE =
  839    0927  2 BEGIN
  840    0928  2 !++
  841    0929  2 !
  842    0930  2 !   FUNCTIONAL DESCRIPTION:
  843    0931  2 !
  844    0932  2 !       Set some fields in the filb using data from the namb
  845    0933  2 !
  846    0934  2 !   INPUTS:
  847    0935  2 !
  848    0936  2 !       namb - address of namb
  849    0937  2 !       filb - address of filb
  850    0938  2 !
  851    0939  2 !   IMPLICIT INPUTS:
  852    0940  2 !
  853    0941  2 !       none
  854    0942  2 !
  855    0943  2 !   OUTPUTS:
  856    0944  2 !
  857    0945  2 !       none
  858    0946  2 !
  859    0947  2 !   IMPLICIT OUTPUTS:
  860    0948  2 !
  861    0949  2 !       none
  862    0950  2 !
  863    0951  2 !   ROUTINE VALUE:
  864    0952  2 !
  865    0953  2 !       none
  866    0954  2 !
  867    0955  2 !   SIDE EFFECTS:
  868    0956  2 !
  869    0957  2 !       none
  870    0958  2 !--
  871    0959  2
  872    0960  2 $dbgtrc_prefix ('copy_namb_to_filb> ');
  873    0961  2
  874    0962  2 $block_check (2, .namb, namb, 523);
  875    0963  2 $block_check (2, .filb, filb, 524);
  876    0964  2
  877    0965  2 ! Set fields in the file context block
  878    0966  2
  879    0967  2 filb [filb$a_assoc_namb]       = .namb;                            ! Pointer to the namb
  880    0968  2 filb [filb$a_assoc_volb]       = .namb [namb$a_assoc_volb];        ! Pointer to the volb (0 if Files-11
  881    0969  2 filb [filb$b_car_control]      = .namb [namb$b_car_control];       ! Carriage control byte
  882    0970  2 filb [filb$b_rec_format]       = .namb [namb$b_rec_format];        ! Record format byte
  883    0971  2 filb [filb$b_transfer_mode]    = .namb [namb$b_transfer_mode];     ! Transfer mode byte
  884    0972  2 filb [filb$l_fixed_len]        = .namb [namb$l_fixed_len];         ! Record length (format=fixed only)
  885    0973  2 filb [filb$b_pad_char]         = .namb [namb$b_pad_char];          ! Pad character (format=fixed only)
  886    0974  2 filb [filb$v_rfmt_explicit]    = .namb [namb$v_rfmt_explicit];     ! A /RECORD was seen
  887    0975  2 filb [filb$v_cctl_explicit]    = .namb [namb$v_cctl_explicit];     ! A /CARRIAGE was seen
  888    0976  2 filb [filb$v_explicit_version] = .namb [namb$v_explicit_version];  ! Explicit version number specified
  889    0977  2
  890    0978  2 ! Virtual devices will have meaningless vol_formats in the namb.  Copy the volb format to the namb always.
  891    0979  2 !
  892    0980  3 IF (.filb [filb$a_assoc_volb] NEQ 0)
  893    0981  2 THEN
```

```
  894   0982  3        BEGIN
  895   0983  3        BIND
  896   0984  3            volb = filb [filb$a_assoc_volb] : $ref_bblock;
  897   0985  3        namb [namb$b_vol_format] = .volb [volb$b_vol_format];
  898   0986  2        END;
  899   0987  2
  900   0988  2 RETURN;
  901   0989  1 END;


                                          003C 00000      .ENTRY    EXCH$COPY_NAMB_TO_FILB, Save R2,R3,R4,R5   ; 0925
                       55 00000000G EF 9E 00002      MOVAB     EXCH$UTIL_BLOCK_CHECK, R5
                       54       04 AC D0 00009      MOVL      NAMB, R4                                       ; 0962
                       52 010A00F7 8F D0 0000D      MOVL      #17432823, R2
                       51       020B 8F 3C 00014      MOVZWL    #523, R1
                       50          54 D0 00019      MOVL      R4, R0
                                   65 16 0001C      JSB       EXCH$UTIL_BLOCK_CHECK
                       53       08 AC D0 0001E      MOVL      FILB, R3                                        ; 0963
                       52 035B00FA 8F D0 00022      MOVL      #56295674, R2
                       51       020C 8F 3C 00029      MOVZWL    #524, R1
                       50          53 D0 0002E      MOVL      R3, R0
                                   65 16 00031      JSB       EXCH$UTIL_BLOCK_CHECK
                    18 A3          54 D0 00033      MOVL      R4, 24(R3)                                      ; 0967
                    1C A3       74 A4 D0 00037      MOVL      116(R4), 28(R3)                                 ; 0968
                    28 A3       7B A4 90 0003C      MOVB      123(R4), 40(R3)                                 ; 0970
                    29 A3       7C A4 B0 00041      MOVW      124(R4), 41(R3)                                 ; 0971
                    35 A3       7E A4 D0 00046      MOVL      126(R4), 53(R3)                                 ; 0972
                    39 A3     0082 C4 90 0004B      MOVB      130(R4), 57(R3)                                 ; 0973
     2B A3       01 00     0085 C4 F0 00051      INSV      133(R4), #0, #1, 43(R3)                         ; 0974
        50     0085 C4       01 EF 00059      EXTZV     #1, #1, 133(R4), R0                             ; 0975
     2B A3       01          50 F0 00060      INSV      R0, #1, #1, 43(R3)
        50       6D A4       03 EF 00066      EXTZV     #3, #1, 109(R4), R0                             ; 0976
     2B A3       01          05 50 F0 0006C      INSV      R0, #5, #1, 43(R3)
                          1C A3 D5 00072      TSTL      28(R3)                                          ; 0980
                          09 13 00075      BEQL      1$
                 50       1C A3 D0 00077      MOVL      28(R3), R0                                       ; 0985
              7A A4       58 A0 90 0007B      MOVB      88(R0), 122(R4)
                             04 00080 1$:    RET                                                          ; 0989
```

; Routine Size: 129 bytes, Routine Base: EXCH$COPY_CODE + 0633

```
 903   0990  1 GLOBAL ROUTINE copy_output_cleanup : NOVALUE =  %SBTTL 'copy_output_cleanup'
 904   0991  2 BEGIN
 905   0992  2 !++
 906   0993  2 !
 907   0994  2 ! FUNCTIONAL DESCRIPTION:
 908   0995  2 !
 909   0996  2 !     Clean up the output file info.  Release the namb and other structures.
 910   0997  2 !
 911   0998  2 ! INPUTS:
 912   0999  2 !
 913   1000  2 !     none
 914   1001  2 !
 915   1002  2 ! IMPLICIT INPUTS:
 916   1003  2 !
 917   1004  2 !     copy$a_out_filb field in copy work area
 918   1005  2 !     copy$a_out_namb field in copy work area
 919   1006  2 !
 920   1007  2 ! OUTPUTS:
 921   1008  2 !
 922   1009  2 !     none
 923   1010  2 !
 924   1011  2 ! IMPLICIT OUTPUTS:
 925   1012  2 !
 926   1013  2 !     none
 927   1014  2 !
 928   1015  2 ! ROUTINE VALUE:
 929   1016  2 !
 930   1017  2 !     none
 931   1018  2 !
 932   1019  2 ! SIDE EFFECTS:
 933   1020  2 !
 934   1021  2 !     none
 935   1022  2 !--
 936   1023  2
 937   1024  2 $dbgtrc_prefix ('copy_output_cleanup> ');
 938   1025  2
 939   1026  2
 940   1027  2 BIND
 941   1028  2     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
 942   1029  2     out_filb = copy [copy$a_out_filb]    : $ref_bblock, ! Filb for the output
 943   1030  2     out_namb = copy [copy$a_out_namb]    : $ref_bblock, ! Namb for the output
 944   1031  2     ctx = out_filb [filb$a_context]      : $ref_bblock  ! Volume specific context
 945   1032  2     ;
```

```
 947      1033   2  ! If a context block is present release it
 948      1034   2  !
 949      1035   2  IF .ctx NEQ 0
 950      1036   2  THEN
 951      1037   2      CASE .out_namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
 952      1038   2      SET
 953      1039   2          [volb$k_vfmt_dos11] :              exch$util_dos11ctx_release (.ctx);
 954      1040   2          [volb$k_vfmt_files11] :            exch$util_rmsb_release (.ctx);
 955      1041   2          [volb$k_vfmt_rt11] :               exch$util_rt11ctx_release (.ctx);
 956      1042   2          [OUTRANGE,INRANGE] :               $logic_check (0, ?false), 234);
 957      1043   2      TES;
 958      1044   2
 959      1045   2  ! If the output volume was RT-11, flush the directory of any modified segments
 960      1046   2
 961      1047   2  CASE .out_namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
 962      1048   2      SET
 963      1049   2
 964      1050   2      [volb$k_vfmt_rt11] :
 965      1051   2                      BEGIN
 966      1052   2                      BIND
 967      1053   2                          volb = out_filb [filb$a_assoc_volb] : $ref_bblock;
 968      1054   2                      exch$rt11_write_cleanup (.volb);    ! Do sundries necessary before we stop copying
 969      1055   2                      END;
 970      1056   2
 971      1057   2      [INRANGE, OUTRANGE] :
 972      1058   2                      ;                                   ! Nothing to do for these guys
 973      1059   2      TES;
 974      1060   2
 975      1061   2  ! Release the output namb
 976      1062   2  !
 977      1063   2  exch$util_namb_release (.out_namb);
 978      1064   2
 979      1065   2  ! Release the output filb
 980      1066   2  !
 981      1067   2  exch$util_filb_release (.out_filb);
 982      1068   2
 983      1069   2  RETURN;
 984      1070   1  END;
```

```
                                          000C 00000        .ENTRY  COPY_OUTPUT_CLEANUP, Save R2,R3         0990
        50 00000000G  EF           04  C1 00002        ADDL3   #4, EXCH$A_GBL, R0                      1028
        53           60 00000044   8F  C1 0000A        ADDL3   #68, (R0), R3                          1029
        52           60 00000048   8F  C1 00012        ADDL3   #72, (R0), R2                          1030
        51           63           20  C1 0001A        ADDL3   #32, (R3), R1                          1031
                                   61  D5 0001E        TSTL    (R1)                                   1035
                                   44  13 00020        BEQL    6$
                     50            62  D0 00022        MOVL    (R2), R0                               1037
        03           00        7A  A0  8F 00025        CASEB   122(R0), #0, #3
0033         0028         001D         0008     0002A 1$:  .WORD   2$-1$,-
                                                            3$-1$,-
                                                            4$-1$,-
                                                            5$-1$
                     7E       EA  8F  9A 00032 2$:  MOVZBL  #234, -(SP)                            1042
```

```
                                             01 DD 00036            PUSHL    #1
                                 00000000G   BF DD 00038            PUSHL    #EXCH$_BADLOGIC
                      00000000G  00           03 FB 0003E           CALLS    #3, LIB$STOP
                                               1F 11 00045          BRB      6$
                      00000000G  EF            61 DD 00047 3$:      PUSHL    (R1)
                                               01 FB 00049          CALLS    #1, EXCH$UTIL_DOS11CTX_RELEASE
                                               14 11 00050          BRB      6$
                      00000000G  EF            61 DD 00052 4$:      PUSHL    (R1)
                                               01 FB 00054          CALLS    #1, EXCH$UTIL_RMSB_RELEASE
                                               09 11 0005B          BRB      6$
                      00000000G  EF            61 DD 0005D 5$:      PUSHL    (R1)
                                               01 FB 0005F          CALLS    #1, EXCH$UTIL_RT11CTX_RELEASE
                                 52            62 DO 00066 6$:      MOVL     (R2), R2
                      03         00     7A A2  8F 00069             CASEB    122(R2), #0, #3
         000A         0017           0017      0017    0006E 7$:    .WORD    9$-7$,-
                                                                             9$-7$,-
                                                                             9$-7$,-
                                                                             8$-7$

                                               0D 11 00076          BRB      9$
                      50                63     1C C1 00078 8$:      ADDL3    #28, (R3), R0
                                               60 DD 0007C          PUSHL    (R0)
                      00000000G  EF            01 FB 0007E          CALLS    #1, EXCH$RT11_WRITE_CLEANUP
                                               52 DD 00085 9$:      PUSHL    R2
                      00000000G  EF            01 FB 00087          CALLS    #1, EXCH$UTIL_NAMB_RELEASE
                                               63 DD 0008E          PUSHL    (R3)
                      00000000G  EF            01 FB 00090          CALLS    #1, EXCH$UTIL_FILB_RELEASE
                                               04 00097             RET
```

```
; Routine Size:  152 bytes,    Routine Base:  EXCH$COPY_CODE + 06B4
```

```
  986    1071  1 GLOBAL ROUTINE copy_output_close =        %SBTTL 'copy_output_close'
  987    1072  2 BEGIN
  988    1073  2 !++
  989    1074  2 !
  990    1075  2 ! FUNCTIONAL DESCRIPTION:
  991    1076  2 !
  992    1077  2 !     Close the output file
  993    1078  2 !
  994    1079  2 ! INPUTS:
  995    1080  2 !
  996    1081  2 !     none
  997    1082  2 !
  998    1083  2 ! IMPLICIT INPUTS:
  999    1084  2 !
 1000    1085  2 !     copy [copy$a_out_filb] describes the file to be closed
 1001    1086  2 !
 1002    1087  2 ! OUTPUTS:
 1003    1088  2 !
 1004    1089  2 !     none
 1005    1090  2 !
 1006    1091  2 ! IMPLICIT OUTPUTS:
 1007    1092  2 !
 1008    1093  2 !     none
 1009    1094  2 !
 1010    1095  2 ! ROUTINE VALUE:
 1011    1096  2 !
 1012    1097  2 !     Success or worst error encountered.
 1013    1098  2 !
 1014    1099  2 ! SIDE EFFECTS:
 1015    1100  2 !
 1016    1101  2 !     none
 1017    1102  2 !--
 1018    1103  2
 1019    1104  2 $dbgtrc_prefix ('copy_output_close> ');
 1020    1105  2
 1021    1106  2 LOCAL
 1022    1107  2     status
 1023    1108  2     ;
 1024    1109  2
 1025    1110  2 BIND
 1026    1111  2     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
 1027    1112  2     out_filb = copy [copy$a_out_filb]    : $ref_bblock  ! Filb for the output
 1028    1113  2     ;
 1029    1114  2
 1030    1115  2 $trace_print_lit ('entry');
 1031    1116  2 $block_check (2, .copy, copy, 514);
 1032    1117  2 $block_check (2, .out_filb, filb, 515);
 1033    1118  2
 1034    1119  2 ! Call the file-specific close routine
 1035    1120  2 !
 1036    1121  2 RETURN (.out_filb [filb$a_close_routine]) (.out_filb);
 1037    1122  2
 1038    1123  1 END;
```

```
                                          003C 00000          .ENTRY   COPY_OUTPUT_CLOSE, Save R2,R3,R4,R5      1071
                           55 00000000G EF 9E 00002          MOVAB    EXCH$UTIL_BLOCK_CHECK, R5
            53 00000000G EF             04 C1 00009          ADDL3    #4, EXCH$A_GBL, R3                       1111
            54                          63 00000044 8F C1 00011          ADDL3    #68, (R3), R4               1112
                                        52 004C00FF 8F D0 00019          MOVL     #4980991, R2                1116
                                        51      0202 8F 3C 00020          MOVZWL   #514, R1
                                        50             63 D0 00025          MOVL     (R3), R0
                                        65             16 00028          JSB      EXCH$UTIL_BLOCK_CHECK
                                        53             64 D0 0002A          MOVL     (R4), R3                 1117
                                        52 035B00FA 8F D0 0002D          MOVL     #56295674, R2
                                        51      0203 8F 3C 00034          MOVZWL   #515, R1
                                        50             53 D0 00039          MOVL     R3, R0
                                        65             16 0003C          JSB      EXCH$UTIL_BLOCK_CHECK
                                        53             DD 0003E          PUSHL    R3                          1121
                        4A B3           01             FB 00040          CALLS    #1, @74(R3)
                                                       04 00044          RET                                 1123
```

; Routine Size: 69 bytes,   Routine Base: EXCH$COPY_CODE + 074C

```
: 1040    1124  1 GLOBAL ROUTINE copy_output_create =        %SBTTL 'copy_output_create'
: 1041    1125  2 BEGIN
: 1042    1126  2 !++
: 1043    1127  2 !
: 1044    1128  2 !   FUNCTIONAL DESCRIPTION:
: 1045    1129  2 !
: 1046    1130  2 !       Open the output file
: 1047    1131  2 !
: 1048    1132  2 !   INPUTS:
: 1049    1133  2 !
: 1050    1134  2 !       none
: 1051    1135  2 !
: 1052    1136  2 !   IMPLICIT INPUTS:
: 1053    1137  2 !
: 1054    1138  2 !       copy [copy$a_out_filb] describes the file to be opened
: 1055    1139  2 !
: 1056    1140  2 !   OUTPUTS:
: 1057    1141  2 !
: 1058    1142  2 !       none
: 1059    1143  2 !
: 1060    1144  2 !   IMPLICIT OUTPUTS:
: 1061    1145  2 !
: 1062    1146  2 !       none
: 1063    1147  2 !
: 1064    1148  2 !   ROUTINE VALUE:
: 1065    1149  2 !
: 1066    1150  2 !       Success or worst error encountered.
: 1067    1151  2 !
: 1068    1152  2 !   SIDE EFFECTS:
: 1069    1153  2 !
: 1070    1154  2 !       none
: 1071    1155  2 !--
: 1072    1156  2
: 1073    1157  2 $dbgtrc_prefix ('copy_output_create> ');
: 1074    1158  2
: 1075    1159  2 LOCAL
: 1076    1160  2     status
: 1077    1161  2     ;
: 1078    1162  2
: 1079    1163  2 BIND
: 1080    1164  2     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
: 1081    1165  2     out_filb = copy [copy$a_out_filb]    : $ref_bblock, ! Filb for the output
: 1082    1166  2     out_namb = copy [copy$a_out_namb]    : $ref_bblock  ! Namb for the output
: 1083    1167  2     ;
: 1084    1168  2
: 1085    1169  2
: 1086    1170  2 $block_check (2, .copy, copy, 516);
: 1087    1171  2 $block_check (2, .out_filb, filb, 517);
: 1088    1172  2 $block_check (2, .out_namb, namb, 518);
```

N 2

EXCH$COPY   copy verb dispatch and misc routines   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 36
V04-000   copy_output_create   5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (14)

```
: 1090       1173  2 ! Perform the volume-specific create operation
: 1091       1174  2 !
: 1092       1175  2 CASE .out_namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
: 1093       1176  2 SET
: 1094       1177  2    [volb$k_vfmt_dos11]           : status = exch$dos11_create_file ();
: 1095       1178  2    [volb$k_vfmt_files11]         : status = exch$fil11_create_file ();
: 1096       1179  2    [volb$k_vfmt_rt11]            : status = exch$rt11_create_file ();
: 1097       1180  2 !\ [volb$k_vfmt_rtmt]            : $exch_signal_stop (exch$_notimplement);
: 1098       1181  2    [OUTRANGE,INRANGE]            : $logic_check (0, (false), 235);
: 1099       1182  2 TES;
: 1100       1183  2
: 1101       1184  2 RETURN .status;
: 1102       1185  1 END;
```

```
                                    007C 00000        .ENTRY  COPY_OUTPUT_CREATE, Save R2,R3,R4,R5,R6    : 1124
                    56 00000000G EF  9E 00002          MOVAB   EXCH$UTIL_BLOCK_CHECK, R6
          53 00000000G EF          04 C1 00009          ADDL3   #4, EXCH$_GBL, R3                         : 1164
          55           63 00000044 8F C1 00011          ADDL3   #68, (R3), R5                            : 1165
          54           63 00000048 8F C1 00019          ADDL3   #72, (R3), R4                            : 1166
                       52 004C00FF 8F D0 00021          MOVL    #4980991, R2                             : 1170
                       51       0204 8F 3C 00028          MOVZWL  #516, R1
                       50            63 D0 0002D          MOVL    (R3), R0
                                     66 16 00030          JSB     EXCH$UTIL_BLOCK_CHECK
                       52 035B00FA 8F D0 00032          MOVL    #56295674, R2                            : 1171
                       51       0205 8F 3C 00039          MOVZWL  #517, R1
                       50            65 D0 0003E          MOVL    (R5), R0
                                     66 16 00041          JSB     EXCH$UTIL_BLOCK_CHECK
                       53            64 D0 00043          MOVL    (R4), R3                                 : 1172
                       52 010A00F7 8F D0 00046          MOVL    #17432823, R2
                       51       0206 8F 3C 0004D          MOVZWL  #518, R1
                       50            53 D0 00052          MOVL    R3, R0
                                     66 16 00055          JSB     EXCH$UTIL_BLOCK_CHECK
                    03        00  7A A3 8F 00057          CASEB   122(R3), #0, #3                          : 1175
   002C        0024      001C      0008     0005C 1$:     .WORD   2$-1$,-
                                                                  3$-1$,-
                                                                  4$-1$,-
                                                                  5$-1$
                       7E     EB 8F 9A 00064 2$:          MOVZBL  #235, -(SP)                              : 1181
                                  01 DD 00068             PUSHL   #1
                       00000000G 8F DD 0006A             PUSHL   #EXCH$_BADLOGIC
          00000000G 00            03 FB 00070             CALLS   #3, LIB$STOP
                                  04    00077             RET
          00000000G EF            00 FB 00078 3$:         CALLS   #0, EXCH$DOS11_CREATE_FILE               : 1177
                                  04    0007F             RET
          00000000G EF            00 FB 00080 4$:         CALLS   #0, EXCH$FIL11_CREATE_FILE               : 1178
                                  04    00087             RET
          00000000G EF            00 FB 00088 5$:         CALLS   #0, EXCH$RT11_CREATE_FILE                : 1179
                                  04    0008F             RET                                             : 1185
```

; Routine Size: 144 bytes,   Routine Base: EXCH$COPY_CODE + 0791

```
: 1104     1186  1  GLOBAL ROUTINE copy_output_delete : NOVALUE =    %SBTTL 'copy_output_delete'
: 1105     1187  2  BEGIN
: 1106     1188  2  !++
: 1107     1189  2  !
: 1108     1190  2  ! FUNCTIONAL DESCRIPTION:
: 1109     1191  2  !
: 1110     1192  2  !     Delete the output file
: 1111     1193  2  !
: 1112     1194  2  ! INPUTS:
: 1113     1195  2  !
: 1114     1196  2  !     none
: 1115     1197  2  !
: 1116     1198  2  ! IMPLICIT INPUTS:
: 1117     1199  2  !
: 1118     1200  2  !     copy [copy$a_out_filb] describes the file to be deleted
: 1119     1201  2  !
: 1120     1202  2  ! OUTPUTS:
: 1121     1203  2  !
: 1122     1204  2  !     none
: 1123     1205  2  !
: 1124     1206  2  ! IMPLICIT OUTPUTS:
: 1125     1207  2  !
: 1126     1208  2  !     none
: 1127     1209  2  !
: 1128     1210  2  ! ROUTINE VALUE:
: 1129     1211  2  !
: 1130     1212  2  !     Success or worst error encountered.
: 1131     1213  2  !
: 1132     1214  2  ! SIDE EFFECTS:
: 1133     1215  2  !
: 1134     1216  2  !     none
: 1135     1217  2  !--
: 1136     1218  2
: 1137     1219  2  $dbgtrc_prefix ('copy_output_delete> ');
: 1138     1220  2
: 1139     1221  2  LOCAL
: 1140     1222  2      status
: 1141     1223  2      ;
: 1142     1224  2
: 1143     1225  2  BIND
: 1144     1226  2      copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
: 1145     1227  2      out_filb = copy [copy$a_out_filb]    : $ref_bblock  ! filb for the output
: 1146     1228  2      ;
: 1147     1229  2
: 1148     1230  2
: 1149     1231  2  $block_check (2, .copy, copy, 558);
: 1150     1232  2  $block_check (2, .out_filb, filb, 559);
: 1151     1233  2
: 1152     1234  2  ! Call the file-specific delete routine
: 1153     1235  2  !
: 1154     1236  2  IF .out_filb [filb$a_delete_routine] NEQ 0
: 1155     1237  2  THEN
: 1156     1238  2      (.out_filb [filb$a_delete_routine]) (.out_filb);
: 1157     1239  2
: 1158     1240  2  RETURN;
: 1159     1241  1  END;
```

```
                                              003C 00000           .ENTRY   COPY_OUTPUT_DELETE, Save R2,R3,R4,R5       : 1186
                              55 00000000G EF 9E 00002           MOVAB    EXLH$UTIL_BLOCK_CHECK, R5
                    53 00000000G EF          04 C1 00009           ADDL3    #4, EXCH$X_GBL, R3                         : 1226
                    54                       63 00000044 8F C1 00011   ADDL3    #68, (R3), R4                         : 1227
                                             52 004C00FF 8F D0 00019   MOVL     #4980991, R2                          : 1231
                                             51       022E 8F 3C 00020   MOVZWL   #558, R1
                                             50          63 D0 00025   MOVL     (R3), R0
                                             65          16 00028   JSB      EXCH$UTIL_BLOCK_CHECK
                    53                       64          D0 0002A   MOVL     (R4), R3                                 : 1232
                                             52 035B00FA 8F D0 0002D   MOVL     #56295674, R2
                                             51       022F 8F 3C 00034   MOVZWL   #559, R1
                                             50          53 D0 00039   MOVL     R3, R0
                                             65          16 0003C   JSB      EXCH$UTIL_BLOCK_CHECK
                                          4E A3 D5 0003E           TSTL     78(R3)                                    : 1236
                                             06 13 00041           BEQL     1$
                                             53 DD 00043           PUSHL    R3                                        : 1238
                    4E  B3                   01 FB 00045           CALLS    #1, @78(R3)
                                             04 00049 1$:          RET                                               : 1241
```

; Routine Size: 74 bytes,   Routine Base: EXCH$COPY_CODE + 0821

```
: 1161    1242  1  GLOBAL ROUTINE copy_parse_cleanup : NOVALUE =   %SBTTL 'copy_parse_cleanup'
: 1162    1243  2  BEGIN
: 1163    1244  2  !++
: 1164    1245  2  !
: 1165    1246  2  !  FUNCTIONAL DESCRIPTION:
: 1166    1247  2  !
: 1167    1248  2  !      Clean up after a successful parse.  Release the namb and other structures.
: 1168    1249  2  !
: 1169    1250  2  !  INPUTS:
: 1170    1251  2  !
: 1171    1252  2  !      none
: 1172    1253  2  !
: 1173    1254  2  !  IMPLICIT INPUTS:
: 1174    1255  2  !
: 1175    1256  2  !      copy$a_inp_namb field in copy work area
: 1176    1257  2  !
: 1177    1258  2  !  OUTPUTS:
: 1178    1259  2  !
: 1179    1260  2  !      none
: 1180    1261  2  !
: 1181    1262  2  !  IMPLICIT OUTPUTS:
: 1182    1263  2  !
: 1183    1264  2  !      none
: 1184    1265  2  !
: 1185    1266  2  !  ROUTINE VALUE:
: 1186    1267  2  !
: 1187    1268  2  !      none
: 1188    1269  2  !
: 1189    1270  2  !  SIDE EFFECTS:
: 1190    1271  2  !
: 1191    1272  2  !      none
: 1192    1273  2  !--
: 1193    1274  2
: 1194    1275  2  $dbgtrc_prefix ('copy_parse_cleanup> ');
: 1195    1276  2
: 1196    1277  2
: 1197    1278  2  BIND
: 1198    1279  2      copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
: 1199    1280  2      inp_filb = copy [copy$a_inp_filb]     : $ref_bblock, ! Filb for the input
: 1200    1281  2      inp_namb = copy [copy$a_inp_namb]     : $ref_bblock, ! Namb for the input
: 1201    1282  2      ctx = inp_filb [filb$a_context]       : $ref_bblock  ! Volume specific context
: 1202    1283  2      ;
```

```
 1204   1284  2  ! If a context block is present release it
 1205   1285  2  !
 1206   1286  2  IF .ctx NEQ 0
 1207   1287  2  THEN
 1208   1288  2      CASE .inp_namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
 1209   1289  2      SET
 1210   1290  2          [volb$k_vfmt_dos11] :              exch$util_dos11ctx_release (.ctx);
 1211   1291  2          [volb$k_vfmt_files11] :            exch$util_rmsb_release (.ctx);
 1212   1292  2          [volb$k_vfmt_rt11] :               exch$util_rt11ctx_release (.ctx);
 1213   1293  2          [OUTRANGE,INRANGE] :               $logic_check (0, ?false), 236);
 1214   1294  2      TES;
 1215   1295  2
 1216   1296  2  ! Release the input namb
 1217   1297  2  !
 1218   1298  2  exch$util_namb_release (.inp_namb);
 1219   1299  2
 1220   1300  2  ! Release the input filb
 1221   1301  2  !
 1222   1302  2  exch$util_filb_release (.inp_filb);
 1223   1303  2
 1224   1304  2  RETURN;
 1225   1305  1  END;
```

```
                                    000C 00000        .ENTRY  COPY_PARSE_CLEANUP, Save R2,R3
          50 00000000G EF           04 C1 00002        ADDL3   #4, EXCH$A_GBL, R0
          53          60            3C C1 0000A        ADDL3   #60, (R0), R3
          52          60 00000040   8F C1 0000E        ADDL3   #64, (R0), R2
          51          63            20 C1 00016        ADDL3   #32, (R3), R1
                                    61 D5 0001A        TSTL    (R1)
                                    44 13 0001C        BEQL    6$
          50                        62 D0 0001E        MOVL    (R2), R0
   0033   0028   03  00  7A  A0  8F 00021              CASEB   122(R0), #0, #3
                      001D    0008  00026 1$:  .WORD   2$-1$,-
                                                       3$-1$,-
                                                       4$-1$,-
                                                       5$-1$
                      7E     EC  8F  9A 0002E 2$:      MOVZBL  #236, -(SP)
                             01  DD 00032              PUSHL   #1
                   00000000G 8F  DD 00034              PUSHL   #EXCH$_BADLOGIC
          00000000G 00        03  FB 0003A              CALLS   #3, LIB$STOP
                             1F  11 00041              BRB     6$
                             61  DD 00043 3$:          PUSHL   (R1)
          00000000G EF        01  FB 00045              CALLS   #1, EXCH$UTIL_DOS11CTX_RELEASE
                             14  11 0004C              BRB     6$
                             61  DD 0004E 4$:          PUSHL   (R1)
          00000000G EF        01  FB 00050              CALLS   #1, EXCH$UTIL_RMSB_RELEASE
                             09  11 00057              BRB     6$
                             61  DD 00059 5$:          PUSHL   (R1)
          00000000G EF        01  FB 0005B              CALLS   #1, EXCH$UTIL_RT11CTX_RELEASE
                             62  DD 00062 6$:          PUSHL   (R2)
          00000000G EF        01  FB 00064              CALLS   #1, EXCH$UTIL_NAMB_RELEASE
                             63  DD 0006B              PUSHL   (R3)
          00000000G EF        01  FB 0006D              CALLS   #1, EXCH$UTIL_FILB_RELEASE
```

1242
1279
1280
1281
1282
1286

1288

1293

1290

1291

1292
1298

1302

                                                          04 00074   RET   : 1305

; Routine Size: 117 bytes,   Routine Base: EXCH$COPY_CODE + 086B

EXCH$COPY
V04-000

copy verb dispatch and misc routines
copy_parse_next_input

G 3
16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742
5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1

Page 42
(18)

```
1227    1306    1    GLOBAL ROUTINE copy_parse_next_input =  %SBTTL 'copy_parse_next_input'
1228    1307    2    BEGIN
1229    1308    2    !++
1230    1309    2    !
1231    1310    2    !  FUNCTIONAL DESCRIPTION:
1232    1311    2    !
1233    1312    2    !        Fetch the next input parameter.  Parse the filename and initialize the input file work region.
1234    1313    2    !  INPUTS:
1235    1314    2    !
1236    1315    2    !        none
1237    1316    2    !
1238    1317    2    !  IMPLICIT INPUTS:
1239    1318    2    !
1240    1319    2    !        Command qualifier value as returned from CLI$xxx routines.  COPY command work area.
1241    1320    2    !
1242    1321    2    !  OUTPUTS:
1243    1322    2    !
1244    1323    2    !        none
1245    1324    2    !
1246    1325    2    !  IMPLICIT OUTPUTS:
1247    1326    2    !
1248    1327    2    !        Command work area receives parse info
1249    1328    2    !
1250    1329    2    !  ROUTINE VALUE:
1251    1330    2    !
1252    1331    2    !        Success or worst error encountered.
1253    1332    2    !
1254    1333    2    !  SIDE EFFECTS:
1255    1334    2    !
1256    1335    2    !        none
1257    1336    2    !
1258    1337    2    !--
1259    1338    2
1260    1339    2    $dbgtrc_prefix ('copy_parse_next_input> ');
1261    1340    2
1262    1341    2    LOCAL
1263    1342    2        status
1264    1343    2        ;
1265    1344    2
1266    1345    2    BIND
1267    1346    2        copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
1268    1347    2        inp_filb = copy [copy$a_inp_filb]     : $ref_bblock, ! Filb for the input
1269    1348    2        inp_namb = copy [copy$a_inp_namb]     : $ref_bblock  ! Namb for the input
1270    1349    2        ;
1271    1350    2
1272    1351    2
1273    1352    2    $block_check (2, .copy, copy, 412);
1274    1353    2
1275    1354    2    ! Fetch the filename and a pointer to a namb
1276    1355    2
1277    1356    3    IF NOT (status = exch$cmd_parse_filespec (%ASCID 'INPUT', copy [copy$q_input_sticky_name], 0,
1278    1357    3                                copy [copy$q_input_filename], inp_namb))
1279    1358    2    THEN
1280    1359    3        BEGIN
1281    1360    3        IF .status NEQ 0
1282    1361    3        THEN
1283    1362    3            $exch_signal (exch$_parseerr, 1, copy [copy$q_input_filename], .status);
```

```
1284   1363   3        RETURN .status;                                    ! No more files to copy, or error in parse
1285   1364   2        END;
1286   1365   2      $debug_print_fao ('input parameter is "!AS"', copy [copy$q_input_filename]);
1287   1366
1288   1367   2      ! If if the input potentially describes multiple files, then set the bit
1289   1368   2      !
1290   1369   2      IF .inp_namb [namb$v_more_files] OR .inp_namb [namb$v_wildcard]
1291   1370   2      THEN
1292   1371   2          copy [copy$v_multiple_files] = true;
1293   1372
1294   1373   2      ! If a foreign device is not mounted, then perform an implied mount
1295   1374   2      !
1296   1375   2      IF    (.inp_namb [namb$a_assoc_volb] EQL 0)
1297   1376   2        AND
1298   1377   4          (BEGIN
1299   1378   4           BIND
1300   1379   5            dev = inp_namb [namb$l_fabdev] : $bblock;
1301   1380   5           .dev [dev$v_for] OR (NOT (.dev [dev$v_mnt]))
1302   1381   2           END)
1303   1382   2        AND
1304   1383   4          ((.inp_namb [namb$b_devclass] EQL dc$_disk)
1305   1384   3           OR
1306   1385   3           (.inp_namb [namb$b_devclass] EQL dc$_tape))
1307   1386   2      THEN
1308   1387   3          BEGIN
1309   1388   3
1310   1389   4          IF NOT (status = exch$moun_implied_mount (.inp_namb))
1311   1390   3          THEN
1312   1391   4              BEGIN
1313   1392   4              exch$util_namb_release (.inp_namb);
1314   1393   4              RETURN .status;
1315   1394   3              END;
1316   1395
1317   1396   3          ! We should now have a valid volb, but we still should check
1318   1397   3          !
1319   1398   3          $block_check (2, .inp_namb [namb$a_assoc_volb], volb, 413);
1320   1399   3
1321   1400   2          END;
1322   1401   2
1323   1402   2      ! Now copy the full name to the default name for proper stickiness
1324   1403   2      !
1325   1404   2      str$copy_dx (copy [copy$q_input_sticky_name], inp_namb [namb$q_fullname]);
1326   1405   2
1327   1406   2      ! Allocate a file block to contain the input information
1328   1407   2      !
1329   1408   2      inp_filb = exch$util_filb_allocate ();
1330   1409   2      exch$copy_namb_to_filb (.inp_namb, .inp_filb);  ! Copy from the namb to the filb
1331   1410   2
1332   1411   2      ! Refetch the positional REWIND qualifier.  DOS-11 clears this bit after rewinding the tape, therefore we mu
1333   1412   2      ! set it once for each parameter
1334   1413   2      !
1335   1414   2      copy [copy$v_q_rewind] = cli$present (%ASCID 'REWIND');
1336   1415   2
1337   1416   2      ! We allow several "output" qualifiers to be on the input filespec.  We interpret "output" quals on the outp
1338   1417   2      ! spec as applying to all output files, whereas "output" quals on the input spec apply only to files created
1339   1418   2      ! this input spec.
1340   1419   2      !
```

```
1341    1420  2 IF NOT .copy [copy$v_type_command]
1342    1421  2 THEN
1343    1422  3     BEGIN
1344    1423  3     inp_filb [filb$v_q_best_try_contiguous]       = cli$present (ascid_best_try);
1345    1424  3     inp_filb [filb$v_q_contiguous]                = cli$present (ascid_contiguous);
1346    1425  3     inp_filb [filb$v_q_truncate]                  = cli$present (ascid_truncate);
1347    1426  3
1348    1427  3     ! Get integer-valued "output" qualifiers, routine signals on errors.  If the qualifier is not present, 0
1349    1428  3     ! in the second parameter and -1 (success) is returned as the routine value.
1350    1429  3
1351    1430  4     IF NOT (status = exch$cmd_cli_get_integer (ascid_allocation, inp_filb [filb$l_q_allocation]))
1352    1431  3     THEN
1353    1432  4         BEGIN
1354    1433  4         copy_parse_cleanup ();
1355    1434  4         RETURN .status;
1356    1435  3         END;
1357    1436  4     IF NOT (status = exch$cmd_cli_get_integer (ascid_extension, inp_filb [filb$l_q_extension]))
1358    1437  3     THEN
1359    1438  4         BEGIN
1360    1439  4         copy_parse_cleanup ();
1361    1440  4         RETURN .status;
1362    1441  3         END;
1363    1442  2     END;
1364    1443  2
1365    1444  2 RETURN .status;
1366    1445  1 END;



                                          .PSECT   EXCH$COPY_PLIT,NOWRT,2

        00 00 00 54 55 50 4E 49  00100 P.ABF:  .ASCII   \INPUT\<0><0><0>
                     010E0005  00108 P.ABE:  .LONG    17694725
                     00000000' 0010C          .ADDRESS P.ABF
        00 00 44 4E 49 57 45 52  00110 P.ABH:  .ASCII   \REWIND\<0><0>
                     010E0006  00118 P.ABG:  .LONG    17694726
                     00000000' 0011C          .ADDRESS P.ABH


                                          .PSECT   EXCH$COPY_CODE,NOWRT,2

                     01FC 00000          .ENTRY   COPY_PARSE_NEXT_INPUT, Save R2,R3,R4,R5,R6,-; 1306
                                                  R7,R8
        58 00000000G EF 9E 00002          MOVAB    EXCH$CMD_CLI_GET_INTEGER, R8
        57 00000000G EF 9E 00009          MOVAB    EXCH$UTIL_BLOCK_CHECK, R7
        56 00000000G 00 9E 00010          MOVAB    CLI$PRESENT, R6
     50 00000000G EF    04 C1 00017        ADDL3    #4, EXCH$A_GBL, R0                       1346
                    54    60 D0 0001F        MOVL     (R0), R4                                1347
                    52 004C00FF 8F D0 00022  MOVL     #4980991, R2                            1352
                    51    019C 8F 3C 00029    MOVZWL   #412, R1
                    50          54 D0 0002E    MOVL     R4, R0
                                67 16 00031    JSB      EXCH$UTIL_BLOCK_CHECK
                          40 A4 9F 00033      PUSHAB   64(R4)                                1357
                          0C A4 9F 00036      PUSHAB   12(R4)
                             7E D4 00059      CLRL     -(SP)
                          1C A4 9F 0003B      PUSHAB   28(R4)                                1356
```

```
                                     0000'  CF  9F  0003E              PUSHAB    P.ABE
                   00000000G  EF             05  FB  00042             CALLS     #5, EXCH$CMD_PARSE_FILESPEC               1357
                              55                  50  D0  00049        MOVL      R0, STATUS
                              18                  55  E8  0004C        BLBS      STATUS, 1$
                                                  59  13  0004F        BEQL      6$                                       1360
                                                  55  DD  00051        PUSHL     STATUS                                   1362
                                        0C  A4    9F  00053            PUSHAB    12(R4)
                                        01  DD    00056                PUSHL     #1
                            00000000G   8F  DD    00058                PUSHL     #EXCH$_PARSEERR
                   00000000G  00         04  FB  0005E                 CALLS     #4, LIB$SIGNAL
                                                  43  11  00065        BRB       6$                                       1363
                              53         40  A4   D0  00067  1$:       MOVL      64(R4), R3                               1369
                              6D         A3  95  0006B                 TSTB      109(R3)
                                         04  19  0006E                 BLSS      2$
                              04  6C     A3  E9  00070                 BLBC      108(R3), 3$
                          34  A4         01  88  00074  2$:            BISB2     #1, 52(R4)                               1371
                              74         A3  D5  00078  3$:            TSTL      116(R3)                                  1375
                                         42  12  0007B                 BNEQ      8$
                          05  6B         A3  E8  0007D                 BLBS      107(R3), 4$                              1380
                   39     6A  A3         03  E0  00081                 BBS       #3, 106(R3), 8$
                              01  78     A3  91  00086  4$:            CMPB      120(R3), #1                              1383
                                         06  13  0008A                 BEQL      5$
                              02  78     A3  91  0008C                 CMPB      120(R3), #2                              1385
                                         2D  12  00090                 BNEQ      8$
                              53  DD     00092  5$:                    PUSHL     R3                                       1389
                   00000000G  EF         01  FB  00094                 CALLS     #1, EXCH$MOUN_IMPLIED_MOUNT
                              55             50  D0  0009B             MOVL      R0, STATUS
                              0C             55  E8  0009E             BLBS      STATUS, 7$
                              53  DD     000A1                         PUSHL     R3                                       1392
                   00000000G  EF         01  FB  000A3                 CALLS     #1, EXCH$UTIL_NAMB_RELEASE
                                     0095  31  000AA  6$:              BRW       10$                                     1393
                   52  041B00F3  8F     D0  000AD  7$:                 MOVL      #68878579, R2                            1398
                          51   019D  8F  3C  000B4                     MOVZWL    #413, R1
                              50      74  A3  D0  000B9                MOVL      116(R3), R0
                                         67  16  000BD                 JSB       EXCH$UTIL_BLOCK_CHECK
                              18  A3     9F  000BF  8$:                PUSHAB    24(R3)                                   1404
                              1C  A4     9F  000C2                     PUSHAB    28(R4)
                   00000000G  00         02  FB  000C5                 CALLS     #2, STR$COPY_DX
                   00000000G  EF         00  FB  000CC                 CALLS     #0, EXCH$UTIL_FILB_ALLOCATE             1408
                              3C  A4     50  D0  000D3                 MOVL      R0, 60(R4)
                              52         3C  A4  D0  000D7             MOVL      60(R4), R2                               1409
                              52  DD     000DB                         PUSHL     R2
                              53  DD     000DD                         PUSHL     R3
                   FC6F  CF   02  FB  000DF                            CALLS     #2, EXCH$COPY_NAMB_TO_FILB
                                    0000'  CF  9F  000E4              PUSHAB    P.ABG                                    1414
                              66         01  FB  000E8                 CALLS     #1, CLI$PRESENT
       31  A4       01         50  F0  000EB                           INSV      R0, #0, #1, 49(R4)
                   4C   34  A4         01  E0  000F1                    BBS       #1, 52(R4), 10$                         1420
                                    0000'  CF  9F  000F6              PUSHAB    ASCID_BEST_TRY                           1423
                              66         01  FB  000FA                 CALLS     #1, CLI$PRESENT
       2C  A2       01         00         50  F0  000FD                INSV      R0, #0, #1, 44(R2)
                                    0000'  CF  9F  00103              PUSHAB    ASCID_CONTIGUOUS                         1424
                              66         01  FB  00107                 CALLS     #1, CLI$PRESENT
       2C  A2       01         01         50  F0  0010A                INSV      R0, #1, #1, 44(R2)
                                    0000'  CF  9F  00110              PUSHAB    ASCID_TRUNCATE                           1425
                              66         01  FB  00114                 CALLS     #1, CLI$PRESENT
       2C  A2       01         02         50  F0  00117                INSV      R0, #2, #1, 44(R2)
```

```
                        2D  A2  9F 0011D         PUSHAB  45(R2)                                              : 1430
                    0000'  CF  9F 00120          PUSHAB  ASCID_ALLOCATION                                    :
                   68      02  FB 00124          CALLS   #2, EXCH$CMD_CLI_GET_INTEGER                        :
                   55      50  D0 00127          MOVL    R0, STATUS                                          :
                   10      55  E9 0012A          BLBC    STATUS, 9$                                          :
                        31  A2  9F 0012D         PUSHAB  49(R2)                                              : 1436
                    0000'  CF  9F 00130          PUSHAB  ASCID_EXTENSION                                     :
                   68      02  FB 00134          CALLS   #2, EXCH$CMD_CLI_GET_INTEGER                        :
                   55      50  D0 00137          MOVL    R0, STATUS                                          :
                   05      55  E8 0013A          BLBS    STATUS, 10$                                         :
          FE49  CF        00  FB 0013D 9$:       CALLS   #0, COPY_PARSE_CLEANUP                              : 1439
                   50      55  D0 00142 10$:      MOVL    STATUS, R0                                         : 1444
                        04 00145                 RET                                                         : 1445

; Routine Size:  326 bytes,    Routine Base:  EXCH$COPY_CODE + 08E0
```

EXCH$COPY
V04-000
copy verb dispatch and misc routines
exch$copy_type
L 3
16-Sep-1984 00:41:48
5-Sep-1984 22:04:55
VAX-11 Bliss-32 V4.0-742
[EXCHNG.SRC]EXCCOPY.B32;1
Page 47
(19)

```
: 1368      1446   1 GLOBAL ROUTINE exch$copy_type = %SBTTL 'exch$copy_type'
: 1369      1447   2 BEGIN
: 1370      1448   2 !+
: 1371      1449   2 !
: 1372      1450   2 !   FUNCTIONAL DESCRIPTION:
: 1373      1451   2 !
: 1374      1452   2 !       Action routine for the type verb, parses and performs main control functions for type
: 1375      1453   2 !
: 1376      1454   2 !   INPUTS:
: 1377      1455   2 !
: 1378      1456   2 !       none
: 1379      1457   2 !
: 1380      1458   2 !   IMPLICIT INPUTS:
: 1381      1459   2 !
: 1382      1460   2 !       Command parameters and qualifiers as returned from CLI$ routines.  Global environment ref'd by exch$
: 1383      1461   2 !
: 1384      1462   2 !   OUTPUTS:
: 1385      1463   2 !
: 1386      1464   2 !       none
: 1387      1465   2 !
: 1388      1466   2 !   IMPLICIT OUTPUTS:
: 1389      1467   2 !
: 1390      1468   2 !       none
: 1391      1469   2 !   ROUTINE VALUE:
: 1392      1470   2 !
: 1393      1471   2 !       Success or worst error encountered.
: 1394      1472   2 !
: 1395      1473   2 !   SIDE EFFECTS:
: 1396      1474   2 !
: 1397      1475   2 !
: 1398      1476   2 !       Files may be created.
: 1399      1477   2 !--
: 1400      1478   2
: 1401      1479   2 $dbgtrc_prefix ('copy_type> ');
: 1402      1480   2
: 1403      1481   2 LOCAL
: 1404      1482   2     copy : $ref_bblock,                          ! Pointer to work area
: 1405      1483   2     inp_filb : $ref_bblock,
: 1406      1484   2     status
: 1407      1485   2     ;
: 1408      1486   2
: 1409      1487   2
: 1410      1488   2 ! Allocate and/or initialize the work area
: 1411      1489   2 !
: 1412      1490   2 copy_init ();
: 1413      1491   2
: 1414      1492   2 ! Get pointers that we need.  Have to wait until work area allocated by init call
: 1415      1493   2 !
: 1416      1494   2 copy = .exch$a_gbl [excg$a_copy_work];           ! Pointer to work area
: 1417      1495   2 copy [copy$v_type_command] = true;
: 1418      1496   2
: 1419      1497   2 ! Init the name used for the input file default
: 1420      1498   2 !
: 1421      1499   2 str$copy_dx (copy [copy$q_input_sticky_name], %ASCID '.LIS');
```

EXCH$COPY   copy verb dispatch and misc routines   M 3
V04-000   exch$copy_type   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 48
                                                            5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (20)

```
: 1423      1500  2 ! Loop through the list of input file specifications.  Errors will be signalled.
: 1424      1501  2 !
: 1425      1502  2 status = rms$_fnf;
: 1426      1503  2 WHILE copy_parse_next_input ()                     ! Get next input file parameter
: 1427      1504  2 DO
: 1428      1505  3     BEGIN
: 1429      1506  3
: 1430      1507  3     inp_filb = .copy [copy$a_inp_filb];            ! The input filb
: 1431      1508  3
: 1432      1509  3     WHILE copy_input_open ()                       ! Open the input file, loop for wildcards
: 1433      1510  3     DO
: 1434      1511  4         BEGIN
: 1435      1512  4         REGISTER
: 1436      1513  4             rec_count;
: 1437      1514  4
: 1438      1515  4         ! Print the file name if file list or wildcards
: 1439      1516  4         !
: 1440      1517  4         IF .copy [copy$v_multiple_files]
: 1441      1518  4         THEN
: 1442      1519  5             BEGIN
: 1443      1520  5             REGISTER
: 1444      1521  5                 fao_desc = 0 : $ref_bblock;
: 1445      1522  5             copy_type_print (0, 0);
: 1446      1523  5             fao_desc = exch$util_fao_buffer (%ASCID 'File "!AF"',
: 1447      1524  5                             .inp_filb [filb$l_result_name_len], inp_filb [filb$t_result_name]);
: 1448      1525  5             copy_type_print (.fao_desc [dsc$w_length], .fao_desc [dsc$a_pointer]);
: 1449      1526  5             copy_type_print (0, 0);
: 1450      1527  5             END;
: 1451      1528  4
: 1452      1529  4         ! While we can get records print them on sys$output
: 1453      1530  4         !
: 1454      1531  4         rec_count = 0;
: 1455      1532  4         WHILE (.inp_filb [filb$a_get_routine]) (.inp_filb)
: 1456      1533  4         DO
: 1457      1534  5             BEGIN
: 1458      1535  5             rec_count = .rec_count + 1;
: 1459      1536  5             copy_type_print (.inp_filb [filb$l_record_len], .inp_filb [filb$a_record]);
: 1460      1537  5             IF .exch$a_gbl [excg$v_control_c] THEN EXITLOOP;   ! If we have seen control/c, exit the loop
: 1461      1538  4             END;
: 1462      1539  4
: 1463      1540  4         IF .exch$a_gbl [excg$v_control_c]          ! If control/c, tell them about it
: 1464      1541  4         THEN
: 1465      1542  4             $exch_signal ($info_stat_copy (exch$_canceled))
: 1466      1543  4         ELSE IF .copy [copy$v_q_log]               ! If /LOG, then display file name and count
: 1467      1544  4         THEN
: 1468      1545  4             $exch_signal (exch$_typed, 3, .inp_filb [filb$l_result_name_len], inp_filb [filb$t_result_name],
: 1469      1546  4
: 1470      1547  4         copy_input_close ();
: 1471      1548  4         status = ss$_normal;
: 1472      1549  4         IF .exch$a_gbl [excg$v_control_c] THEN EXITLOOP;
: 1473      1550  3         END;
: 1474      1551  3
: 1475      1552  3     copy_parse_cleanup ();                         ! Release namb, clean up after parse
: 1476      1553  2     IF .exch$a_gbl [excg$v_control_c] THEN EXITLOOP;
: 1477      1554  2     END;
: 1478      1555  2
: 1479      1556  2 RETURN .status;
```

N 3

EXCHSCOPY   copy verb dispatch and misc routines   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 49
V04-000   exch$copy_type   5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (20)

; 1480   1557 1 END;

```
                                                                        .PSECT  EXCH$COPY_PLIT,NOWRT,2

                                        53 49 4C 2E 00120 P.ABJ:         .ASCII  \.LIS\
                                          010E0004  00124 P.ABI:         .LONG   17694724
                                          00000000' 00128               .ADDRESS P.ABJ
            00 00 22 46 41 21 22 20 65 6C 69 46 0012C P.ABL:            .ASCII  \File ''!AF''\<0><0>
                                          010E000A  00138 P.ABK:         .LONG   17694730
                                          00000000' 0013C               .ADDRESS P.ABL

                                                                        .EXTRN  EXCH$_TYPED

                                                                        .PSECT  EXCH$COPY_CODE,NOWRT,2

                                               01FC 00000               .ENTRY  EXCH$COPY_TYPE, Save R2,R3,R4,R5,R6,R7,R8  ; 1446
                        58 00000000G 00 9E 00002                        MOVAB   LIB$SIGNAL, R8
                        57     0000V CF 9E 00009                        MOVAB   COPY_TYPE_PRINT, R7
                        56 U0000000G EF 9E 0000E                        MOVAB   EXCH$A_GBL, R6
                FA62 CF          00 FB 00015                            CALLS   #0, COPY_INIT                              ; 1490
                50               66 D0 0001A                            MOVL    EXCH$A_GBL, R0                             ; 1494
                54           04 A0 D0 0001D                            MOVL    4(R0), COPY
                34 A4           02 88 00021                            BISB2   #2, 52(COPY)                               ; 1495
                           0000' CF 9F 00025                            PUSHAB  P.ABI                                     ; 1499
                              1C A4 9F 00029                            PUSHAB  28(COPY)
                00000000G 00    02 FB 0002C                            CALLS   #2, STR$COPY_DX
                55     00018292 8F D0 00033                            MOVL    #98962, STATUS                            ; 1502
                FE7B CF          00 FB 0003A 1$:                        CALLS   #0, COPY_PARSE_NEXT_INPUT                  ; 1503
                03           50 E8 0003F                            BLBS    R0, 2$
                           0096 31 00042                            BRW     10$
                53          3C A4 D0 00045 2$:                        MOVL    60(COPY), INP_FILB                         ; 1507
                FB33 CF          00 FB 00049 3$:                        CALLS   #0, COPY_INPUT_OPEN                        ; 1509
                7E           50 E9 0004E                            BLBC    R0, 9$
                24 34        A4 E9 00051                            BLBC    52(COPY), 4$                              ; 1517
                7E           7C 00055                            CLRQ    -(SP)                                     ; 1522
                67           02 FB 00057                            CALLS   #2, COPY_TYPE_PRINT
                5A A3        9F 0005A                            PUSHAB  90(INP_FILB)                              ; 1524
                3A A3        DD 0005D                            PUSHL   58(INP_FILB)
                           0000' CF 9F 00060                            PUSHAB  P.ABK                                     ; 1523
                00000000G EF  03 FB 00064                            CALLS   #3, EXCH$UTIL_FAO_BUFFER                   ; 1524
                04 AC        DD 0006B                            PUSHL   4(FAO_DESC)                               ; 1525
                7E           60 3C 0006E                            MOVZWL  (FAO_DESC), -(SP)
                67           02 FB 00071                            CALLS   #2, COPY_TYPE_PRINT
                7E           7C 00074                            CLRQ    -(SP)                                     ; 1526
                67           02 FB 00076                            CALLS   #2, COPY_TYPE_PRINT
                52           D4 00079 4$:                        CLRL    REC_COUNT                                 ; 1531
                53           DD 0007B 5$:                        PUSHL   INP_FILB                                  ; 1532
                52 B3        01 FB 0007D                            CALLS   #1, @82(INP_FILB)
                0D           50 E9 00081                            BLBC    R0, 6$
                52           D6 00084                            INCL    REC_COUNT                                 ; 1535
                7E 42 A3     7D 00086                            MOVQ    66(INP_FILB), -(SP)                        ; 1536
                67           02 FB 0008A                            CALLS   #2, COPY_TYPE_PRINT
                EA 00        B6 E9 0008D                            BLBC    @EXCH$A_GBL, 5$                            ; 1537
                13 00        B6 E9 00091 6$:                        BLBC    @EXCH$A_GBL, 7$                            ; 1540
                50 00000000G 8F D0 00095                            MOVL    #EXCH$_CANCELED, STATUS2                   ; 1542
```

```
          50           03              00              03  F0  0009C          INSV    #3, #0, #3, STATUS2
                                       68                  50  DD  000A1          PUSHL   STATUS2
                                                           01  FB  000A3          CALLS   #1, LIB$SIGNAL
                                                           18  11  000A6          BRB     8$
                       13           30  A4              03  E1  000A8 7$:        BBC     #3, 48(COPY), 8$
                                                           52  DD  000AD          PUSHL   REC_COUNT
                                          5A  A3  9F  000AF          PUSHAB  90(INP_FILB)
                                          3A  A3  DD  000B2          PUSHL   58(INP_FILB)
                                                           03  DD  000B5          PUSHL   #3
                                    00000000G  8F  DD  000B7          PUSHL   #EXCH$_TYPED
                                       68                  05  FB  000BD          CALLS   #5, LIB$SIGNAL
                       FA7B  CF              00  FB  000C0 8$:        CALLS   #0, COPY_INPUT_CLOSE
                                       55                  01  D0  000C5          MOVL    #1, STATUS
                                       03              00  B6  E8  000C8          BLBS    @EXCH$A_GBL, 9$
                                                  FF7A  31  000CC          BRW     3$
                       FD71  CF              00  FB  000CF 9$:        CALLS   #0, COPY_PARSE_CLEANUP
                                       03              00  B6  E8  000D4          BLBS    @EXCH$A_GBL, 10$
                                                  FF5F  31  000D8          BRW     1$
                                       50                  55  D0  000DB 10$:       MOVL    STATUS, R0
                                                           04  000DE          RET
```

```
; Routine Size: 223 bytes,     Routine Base: EXCHSCOPY_CODE + 0A26
```

1543
1545
1547
1548
1549
1552
1553
1556
1557

```
EXCHSCOPY          copy verb dispatch and misc routines              16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742        Page 51
V04-000            copy_type_print (len, rec)                         5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1              (21)
```

```
: 1482    1558  1  GLOBAL ROUTINE copy_type_print (len, rec : $ref_bvector) : NOVALUE =     %SBTTL 'copy_type_print (len, rec)'
: 1483    1559  2  BEGIN
: 1484    1560  2  !++
: 1485    1561  2
: 1486    1562  2  ! FUNCTIONAL DESCRIPTION:
: 1487    1563  2  !
: 1488    1564  2  !     Reformats (non-format control chars replaced by ^char) and prints the record
: 1489    1565  2  !
: 1490    1566  2  ! INPUTS:
: 1491    1567  2  !
: 1492    1568  2  !     len - length of the record to be reformatted
: 1493    1569  2  !     rec - address of the record
: 1494    1570  2  !
: 1495    1571  2  ! IMPLICIT INPUTS:
: 1496    1572  2  !
: 1497    1573  2  !     output rab in global storage
: 1498    1574  2  !
: 1499    1575  2  ! OUTPUTS:
: 1500    1576  2  !
: 1501    1577  2  !     none
: 1502    1578  2  !
: 1503    1579  2  ! IMPLICIT OUTPUTS:
: 1504    1580  2  !
: 1505    1581  2  !     none
: 1506    1582  2  !
: 1507    1583  2  ! ROUTINE VALUE:
: 1508    1584  2  !
: 1509    1585  2  !     none
: 1510    1586  2  !
: 1511    1587  2  ! SIDE EFFECTS:
: 1512    1588  2  !
: 1513    1589  2  !     output on SYS$OUTPUT
: 1514    1590  2  !--
: 1515    1591  2
: 1516    1592  2  $dbgtrc_prefix ('copy_type_print> ');
: 1517    1593  2
: 1518    1594  2  BIND
: 1519    1595  2      copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
: 1520    1596  2      fab = exch$a_gbl [excg$a_sysout_fab] : $ref_bblock, ! Pointer to output fab
: 1521    1597  2      rab = exch$a_gbl [excg$a_sysout_rab] : $ref_bblock  ! Pointer to output rab
: 1522    1598  2      ;
: 1523    1599  2
: 1524    1600  2  ! Define a table of substitute strings for control characters.  We define a byte vector of offsets to
: 1525    1601  2  ! ASCIC substitute strings.  A zero in the table means no substitution, an non-zero is the offset from
: 1526    1602  2  ! the base of the substitute strings.
: 1527    1603  2  !
: 1528    1604  2  BIND
: 1529    1605  2      table_base = UPLIT BYTE (0);
: 1530    1606  2  OWN
: 1531    1607  2      table : VECTOR [256, BYTE] PRESET (
: 1532    1608  2              [%x'00'] =  (UPLIT BYTE (%ascic 'NUL') - table_base),
: 1533    1609  2              [%x'01'] =  (UPLIT BYTE (%ascic 'SOH') - table_base),
: 1534    1610  2              [%x'02'] =  (UPLIT BYTE (%ascic 'STX') - table_base),
: 1535    1611  2              [%x'03'] =  (UPLIT BYTE (%ascic 'ETX') - table_base),
: 1536    1612  2              [%x'04'] =  (UPLIT BYTE (%ascic 'EOT') - table_base),
: 1537    1613  2              [%x'05'] =  (UPLIT BYTE (%ascic 'ENQ') - table_base),
: 1538    1614  2              [%x'06'] =  (UPLIT BYTE (%ascic 'ACK') - table_base),
```

```
1539   1615  2                        [%x'07'] = (UPLIT BYTE (%ascic 'BEL') - table_base),
1540   1616  2                        [%x'08'] = (UPLIT BYTE (%ascic 'BS') - table_base),
1541   1617  2                        [%x'0E'] = (UPLIT BYTE (%ascic 'SO') - table_base),
1542   1618  2                        [%x'0F'] = (UPLIT BYTE (%ascic 'SI') - table_base),
1543   1619  2                        [%x'10'] = (UPLIT BYTE (%ascic 'DLE') - table_base),
1544   1620  2                        [%x'11'] = (UPLIT BYTE (%ascic 'DC1') - table_base),
1545   1621  2                        [%x'12'] = (UPLIT BYTE (%ascic 'DC2') - table_base),
1546   1622  2                        [%x'13'] = (UPLIT BYTE (%ascic 'DC3') - table_base),
1547   1623  2                        [%x'14'] = (UPLIT BYTE (%ascic 'DC4') - table_base),
1548   1624  2                        [%x'15'] = (UPLIT BYTE (%ascic 'NAK') - table_base),
1549   1625  2                        [%x'16'] = (UPLIT BYTE (%ascic 'SYN') - table_base),
1550   1626  2                        [%x'17'] = (UPLIT BYTE (%ascic 'ETB') - table_base),
1551   1627  2                        [%x'18'] = (UPLIT BYTE (%ascic 'CAN') - table_base),
1552   1628  2                        [%x'19'] = (UPLIT BYTE (%ascic 'EM') - table_base),
1553   1629  2                        [%x'1A'] = (UPLIT BYTE (%ascic 'SUB') - table_base),
1554   1630  2            !           [%x'1B'] = (UPLIT BYTE (%ascic 'ESC') - table_base),
1555   1631  2                        [%x'1C'] = (UPLIT BYTE (%ascic 'FS') - table_base),
1556   1632  2                        [%x'1D'] = (UPLIT BYTE (%ascic 'GS') - table_base),
1557   1633  2                        [%x'1E'] = (UPLIT BYTE (%ascic 'RS') - table_base),
1558   1634  2                        [%x'1F'] = (UPLIT BYTE (%ascic 'US') - table_base),
1559   1635  2                        [%x'7F'] = (UPLIT BYTE (%ascic 'DEL') - table_base),
1560   1636  2                        [%x'80'] = (UPLIT BYTE (%ascic 'X80') - table_base),
1561   1637  2                        [%x'81'] = (UPLIT BYTE (%ascic 'X81') - table_base),
1562   1638  2                        [%x'82'] = (UPLIT BYTE (%ascic 'X82') - table_base),
1563   1639  2                        [%x'83'] = (UPLIT BYTE (%ascic 'X83') - table_base),
1564   1640  2                        [%x'84'] = (UPLIT BYTE (%ascic 'IND') - table_base),
1565   1641  2                        [%x'85'] = (UPLIT BYTE (%ascic 'NEL') - table_base),
1566   1642  2                        [%x'86'] = (UPLIT BYTE (%ascic 'SSA') - table_base),
1567   1643  2                        [%x'87'] = (UPLIT BYTE (%ascic 'ESA') - table_base),
1568   1644  2                        [%x'88'] = (UPLIT BYTE (%ascic 'HTS') - table_base),
1569   1645  2                        [%x'89'] = (UPLIT BYTE (%ascic 'HTJ') - table_base),
1570   1646  2                        [%x'8A'] = (UPLIT BYTE (%ascic 'VTS') - table_base),
1571   1647  2                        [%x'8B'] = (UPLIT BYTE (%ascic 'PLD') - table_base),
1572   1648  2                        [%x'8C'] = (UPLIT BYTE (%ascic 'PLU') - table_base),
1573   1649  2                        [%x'8D'] = (UPLIT BYTE (%ascic 'RI') - table_base),
1574   1650  2                        [%x'8E'] = (UPLIT BYTE (%ascic 'SS2') - table_base),
1575   1651  2                        [%x'8F'] = (UPLIT BYTE (%ascic 'SS3') - table_base),
1576   1652  2                        [%x'90'] = (UPLIT BYTE (%ascic 'DCS') - table_base),
1577   1653  2                        [%x'91'] = (UPLIT BYTE (%ascic 'PU1') - table_base),
1578   1654  2                        [%x'92'] = (UPLIT BYTE (%ascic 'PU2') - table_base),
1579   1655  2                        [%x'93'] = (UPLIT BYTE (%ascic 'STS') - table_base),
1580   1656  2                        [%x'94'] = (UPLIT BYTE (%ascic 'CCH') - table_base),
1581   1657  2                        [%x'95'] = (UPLIT BYTE (%ascic 'MW') - table_base),
1582   1658  2                        [%x'96'] = (UPLIT BYTE (%ascic 'SPA') - table_base),
1583   1659  2                        [%x'97'] = (UPLIT BYTE (%ascic 'EPA') - table_base),
1584   1660  2                        [%x'98'] = (UPLIT BYTE (%ascic 'X98') - table_base),
1585   1661  2                        [%x'99'] = (UPLIT BYTE (%ascic 'X99') - table_base),
1586   1662  2                        [%x'9A'] = (UPLIT BYTE (%ascic 'X9A') - table_base),
1587   1663  2                        [%x'9B'] = (UPLIT BYTE (%ascic 'CSI') - table_base),
1588   1664  2                        [%x'9C'] = (UPLIT BYTE (%ascic 'ST') - table_base),
1589   1665  2                        [%x'9D'] = (UPLIT BYTE (%ascic 'OSC') - table_base),
1590   1666  2                        [%x'9E'] = (UPLIT BYTE (%ascic 'PM') - table_base),
1591   1667  2                        [%x'9F'] = (UPLIT BYTE (%ascic 'APC') - table_base),
1592   1668  2                        [%x'A0'] = (UPLIT BYTE (%ascic 'XA0') - table_base),
1593   1669  2                        [%x'FF'] = (UPLIT BYTE (%ascic 'XFF') - table_base);
1594   1670  2  BIND
1595   1671  2          table_top = UPLIT BYTE (0);                    ! Hang a label on the end
```

```
; 1596       1672  2
; 1597       1673  2  ! Make sure that all of the strings total fewer than 256 bytes, so that byte offsets will work.  Note
; 1598       1674  2  ! that BLISS stores the above table like <table-base><ascic-strings><table><table-top> so that we must
; 1599       1675  2  ! include the length of the table itself.  Also test the assumption about storage format.  (We have
; 1600       1676  2  ! defined both OWN and PLIT to the same psect.)
; 1601       1677  2
; 1602    L  1678  2  $logic_check (0, ((table_top-table_base) LEQ 511), 309);
; %PRINT:    assumption 309 verified during compilation
; 1603    L  1679  2  $logic_check (0, ((table_base LSSA table) AND (table LSSA table_top)), 318);
; %PRINT:    assumption 318 verified during compilation
; 1604       1680  2
; 1605       1681  2  LOCAL
; 1606       1682  2      buf : $bvector [filb$s_record_buffer+5],     ! Worst case is buffer of deletes, "<DEL><DEL>..."
; 1607       1683  2      buflen,
; 1608       1684  2      bufptr,
; 1609       1685  2      status
; 1610       1686  2      ;
; 1611       1687  2
; 1612       1688  2  REGISTER
; 1613       1689  2      ip,                                          ! Input pointer
; 1614       1690  2      op;                                          ! Output pointer
```

EXCH$COPY   copy verb dispatch and misc routines   F 4
V04-000   copy_type_print (len, rec)   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 54
                                                              5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (22)

```
: 1616    1691  2 ROUTINE put (len, buf) : NOVALUE =
: 1617    1692  3 BEGIN
: 1618    1693  3
: 1619    1694  3 !+
: 1620    1695  3 ! Local routine to put a record.  If the put fails because the record is too long, we will shorten the
: 1621    1696  3 ! request and try again.
: 1622    1697  3 !-
: 1623    1698  3
: 1624    1699  3 LOCAL
: 1625    1700  3     status;
: 1626    1701  3
: 1627    1702  3 BIND
: 1628    1703  3     copy = exch$a_gbl [excg$a_copy_work] : $ref_bblock, ! Pointer to work area
: 1629    1704  3     fab = exch$a_gbl [excg$a_sysout_fab] : $ref_bblock, ! Pointer to output fab
: 1630    1705  3     rab = exch$a_gbl [excg$a_sysout_rab] : $ref_bblock  ! Pointer to output rab
: 1631    1706  3     ;
: 1632    1707  3
: 1633    1708  3 rab [rab$w_rsz] = .len;
: 1634    1709  3 rab [rab$l_rbf] = .buf;
: 1635    1710  3
: 1636    1711  4 IF NOT (status = $put (rab = .rab))
: 1637    1712  3 THEN
: 1638    1713  4     BEGIN
: 1639    1714  4
: 1640    1715  4     ! If the error is due to a record which was too long, shorten the request and try again
: 1641    1716  4     !
: 1642    1717  5     IF (
: 1643    1718  6             (.copy [copy$l_max_rec] GTRU 80)        ! we aren't pretty short already
: 1644    1719  5         AND
: 1645    1720  6             (
: 1646    1721  7                 (.status EQL RMS$_RSZ)              ! error is rec too big (get from tape)
: 1647    1722  6             OR
: 1648    1723  7                 (
: 1649    1724  8                     (.status EQL RMS$_SYS)         ! terminal maxbuf error
: 1650    1725  7                 AND
: 1651    1726  8                     (.rab [rab$l_stv] EQL SS$_EXQUOTA)
: 1652    1727  7                 )
: 1653    1728  6             )
: 1654    1729  5         )
: 1655    1730  4     THEN
: 1656    1731  5         BEGIN
: 1657    1732  5         copy [copy$l_max_rec] = (.len * 90) / 100;        ! try with rec 90% as long
: 1658    1733  5         put (.copy [copy$l_max_rec], .buf);
: 1659    1734  5         RETURN;
: 1660    1735  5         END
: 1661    1736  4     ELSE
: 1662    1737  4         exch$util_file_error (exch$_writeerr, .status, .fab, .rab [rab$l_stv]);
: 1663    1738  3     END;
: 1664    1739  3
: 1665    1740  3 RETURN;
: 1666    1741  2 END;


                                                  .PSECT  EXCH$COPY_PLIT,NOWRT,2

                                    00  00140 P.ABM:  .BYTE   0                                              ;
```

```
                                4C  55  4E  03    00141 P.ABN:     .ASCII    <3>\NUL\
                                48  4F  53  03    00145 P.ABO:     .ASCII    <3>\SOH\
                                58  54  53  03    00149 P.ABP:     .ASCII    <3>\STX\
                                58  54  45  03    0014D P.ABQ:     .ASCII    <3>\ETX\
                                54  4F  45  03    00151 P.ABR:     .ASCII    <3>\EOT\
                                51  4E  45  03    00155 P.ABS:     .ASCII    <3>\ENQ\
                                4B  43  41  03    00159 P.ABT:     .ASCII    <3>\ACK\
                                4C  45  42  03    0015D P.ABU:     .ASCII    <3>\BEL\
                                    53  42  02    00161 P.ABV:     .ASCII    <2>\BS\
                                    4F  53  02    00164 P.ABW:     .ASCII    <2>\SO\
                                    49  53  02    00167 P.ABX:     .ASCII    <2>\SI\
                                45  4C  44  03    0016A P.ABY:     .ASCII    <3>\DLE\
                                31  43  44  03    0016E P.ABZ:     .ASCII    <3>\DC1\
                                32  43  44  03    00172 P.ACA:     .ASCII    <3>\DC2\
                                33  43  44  03    00176 P.ACB:     .ASCII    <3>\DC3\
                                34  43  44  03    0017A P.ACC:     .ASCII    <3>\DC4\
                                4B  41  4E  03    0017E P.ACD:     .ASCII    <3>\NAK\
                                4E  59  53  03    00182 P.ACE:     .ASCII    <3>\SYN\
                                42  54  45  03    00186 P.ACF:     .ASCII    <3>\ETB\
                                4E  41  43  03    0018A P.ACG:     .ASCII    <3>\CAN\
                                    4D  45  02    0018E P.ACH:     .ASCII    <2>\EM\
                                42  55  53  03    00191 P.ACI:     .ASCII    <3>\SUB\
                                    53  46  02    00195 P.ACJ:     .ASCII    <2>\FS\
                                    53  47  02    00198 P.ACK:     .ASCII    <2>\GS\
                                    53  52  02    0019B P.ACL:     .ASCII    <2>\RS\
                                    53  55  02    0019E P.ACM:     .ASCII    <2>\US\
                                4C  45  44  03    001A1 P.ACN:     .ASCII    <3>\DEL\
                                30  38  58  03    001A5 P.ACO:     .ASCII    <3>\X80\
                                31  38  58  03    001A9 P.ACP:     .ASCII    <3>\X81\
                                32  38  58  03    001AD P.ACQ:     .ASCII    <3>\X82\
                                33  38  58  03    001B1 P.ACR:     .ASCII    <3>\X83\
                                44  4E  49  03    001B5 P.ACS:     .ASCII    <3>\IND\
                                4C  45  4E  03    001B9 P.ACT:     .ASCII    <3>\NEL\
                                41  53  53  03    001BD P.ACU:     .ASCII    <3>\SSA\
                                41  53  45  03    001C1 P.ACV:     .ASCII    <3>\ESA\
                                53  54  48  03    001C5 P.ACW:     .ASCII    <3>\HTS\
                                4A  54  48  03    001C9 P.ACX:     .ASCII    <3>\HTJ\
                                53  54  56  03    001CD P.ACY:     .ASCII    <3>\VTS\
                                44  4C  50  03    001D1 P.ACZ:     .ASCII    <3>\PLD\
                                55  4C  50  03    001D5 P.ADA:     .ASCII    <3>\PLU\
                                    49  52  02    001D9 P.ADB:     .ASCII    <2>\RI\
                                32  53  53  03    001DC P.ADC:     .ASCII    <3>\SS2\
                                33  53  53  03    001E0 P.ADD:     .ASCII    <3>\SS3\
                                53  43  44  03    001E4 P.ADE:     .ASCII    <3>\DCS\
                                31  55  50  03    001E8 P.ADF:     .ASCII    <3>\PU1\
                                32  55  50  03    001EC P.ADG:     .ASCII    <3>\PU2\
                                53  54  53  03    001F0 P.ADH:     .ASCII    <3>\STS\
                                48  43  43  03    001F4 P.ADI:     .ASCII    <3>\CCH\
                                    57  4D  02    001F8 P.ADJ:     .ASCII    <2>\MW\
                                41  50  53  03    001FB P.ADK:     .ASCII    <3>\SPA\
                                41  50  45  03    001FF P.ADL:     .ASCII    <3>\EPA\
                                38  39  58  03    00203 P.ADM:     .ASCII    <3>\X98\
                                39  39  58  03    00207 P.ADN:     .ASCII    <3>\X99\
                                41  39  58  03    0020B P.ADO:     .ASCII    <3>\X9A\
                                49  53  43  03    0020F P.ADP:     .ASCII    <3>\CSI\
                                    54  53  02    00213 P.ADQ:     .ASCII    <2>\ST\
                                43  53  4F  03    00216 P.ADR:     .ASCII    <3>\OSC\
```

EXCHSCOPY   copy verb dispatch and misc routines   H 4
V04-000   copy_type_print (len, rec)   16-Sep-1984 00:41:48   VAX-11 Bliss-32 V4.0-742   Page 56
                                                        5-Sep-1984 22:04:55   [EXCHNG.SRC]EXCCOPY.B32;1   (22)

```
                                      4D  50  02  0021A  P.ADS:  .ASCII    <2>\PM\
                              43  50  41  03  0021D  P.ADT:  .ASCII    <3>\APC\
                              30  41  58  03  00221  P.ADU:  .ASCII    <3>\XA0\
                              46  46  58  03  00225  P.ADV:  .ASCII    <3>\XFF\
                                              00229          .BLKB     3
                  21  1D  19  15  11  0D  09  05  01  0022C  TABLE:  .BYTE     1, 5, 9, 13, 17, 21, 25, 29, 33
                                              00#  00235          .BYTE     0[5]
      51  4E  4A  46  42  3E  3A  36  32  2E  2A  27  24  0023A          .BYTE     36, 39, 42, 46, 50, 54, 58, 62, 66, 70, -
                                                                          74, 78, 81
                                      5E  5B  58  55  00247          .BYTE     0
                                              00#  00248          .BYTE     85, 88, 91, 94
                                              00#  0024C          .BYTE     0[95]
 99  95  91  8D  89  85  81  7D  79  75  71  6D  69  65  61  9C  002AB          .BYTE     97, 101, 105, 109, 113, 117, 121, 125, -
 D3  CF  CB  C7  C3  BF  BB  B8  B4  B0  AC  A8  A4  A0  002BA                          -127, -123, -119, -115, -111, -107, -103, -
                                      E1  DD  DA  D6  002C9                          -100, -96, -92, -88, -84, -80, -76, -72, -
                                                                          -69, -65, -61, -57, -53, -49, -45, -42, -
                                                                          -38, -35, -31
                                              00#  002CD          .BYTE     0[94]
                                              E5  0032B          .BYTE     -27
                                              00  0032C  P.ADW:  .BYTE     0

                                              TABLE_BASE=        P.ABM
                                              TABLE_TOP=         P.ADW
                                                      .EXTRN    SYS$PUT

                                                      .PSECT    EXCH$COPY_CODE,NOWRT,2

                                      001C  00000  PUT:    .WORD     Save R2,R3,R4
                          50  00000000G  EF  D0  00002          MOVL      EXCH$A_GBL, R0
                          53          04  A0  9E  00009          MOVAB     4(R0), R3
                          54        00D0  C0  9E  0000D          MOVAB     208(R0), R4
                          50        00D4  C0  9E  00012          MOVAB     212(R0), R0
                          52          60  D0  00017          MOVL      (R0), R2
                          22  A2      04  AC  B0  0001A          MOVW      LEN, 34(R2)
                          28  A2      08  AC  D0  0001F          MOVL      BUF, 40(R2)
                                      52  DD  00024          PUSHL     R2
              00000000G  00      01  FB  00026          CALLS     #1, SYS$PUT
                          56      50  E8  0002D          BLBS      STATUS, 3$
                          51      63  D0  00030          MOVL      (R3), R1
              00000050  8F      38  A1  D1  00033          CMPL      56(R1), #80
                          35      1B  0003B          BLEQU     2$
              000186A4  8F      50  D1  0003D          CMPL      STATUS, #100004
                          0F      13  00044          BEQL      1$
              0001C10C  BF      50  D1  00046          CMPL      STATUS, #114956
                          23      12  0004D          BNEQ      2$
                      1C      0C  A2  D1  0004F          CMPL      12(R2), #28
                          1D      12  00053          BNEQ      2$
          38      53      04  AC  0000005A  8F  C5  00055  1$:    MULL3     #90, LEN, R3
          38  A1      53  00000064  8F  C7  0005E          DIVL3     #100, R3, 56(R1)
                          08  AC  DD  00067          PUSHL     BUF
                          38  A1  DD  0006A          PUSHL     56(R1)
                  BF  AF      02  FB  0006D          CALLS     #2, PUT
                          04  00071          RET
                      0C  A2  DD  00072  2$:    PUSHL     12(R2)
                          64  DD  00075          PUSHL     (R4)
                          50  DD  00077          PUSHL     STATUS
              00F810D0  8F  DD  00079          PUSHL     #16257232
```

```
                   00000000G  EF                   04  FB 0007F               CALLS   #4, EXCH$UTIL_FILE_ERROR
                                                    04 00086 3$:              RET                                                    ; 1741
```

; Routine Size: 135 bytes,   Routine Base: EXCH$COPY_CODE + 0B05

```
1668    1742   2 ip = .rec;                                          ! Input buffer pointer
1669    1743   2 op = buf;
1670    1744
1671    1745   2 DECR count FROM .len-1 TO 0                          ! Convert the controls
1672    1746   2 DO
1673    1747       BEGIN
1674    1748       REGISTER
1675    1749           char,                                         ! Local character variable
1676    1750           string : $ref_bvector;                        ! Pointer to string for expansion
1677    1751
1678    1752       char = CH$RCHAR_A (ip);                            ! Get next character
1679    1753       IF (string = .table [.char]) NEQ 0                 ! See if the substitution offset is zero
1680    1754       THEN
1681    1755   4       BEGIN
1682    1756   4       REGISTER
1683    1757   4           len;
1684    1758   4       string = .string + table_base;                ! Turn the offset into an address
1685    1759   4       CH$WCHAR_A ('<', op);                          ! Start with the open bracket
1686    1760   4       len = .string [0];                            ! Move the length to a register
1687    1761   4       CH$MOVE (.len, string [1], .op);              ! Copy the ASCIC string
1688    1762   4       op = .op+.len;                                ! Move the output pointer
1689    1763   4       CH$WCHAR_A ('>', op);                          ! And finish with the close bracket
1690    1764   4       END
1691    1765   3   ELSE                                              ! Offset is zero, just move the char
1692    1766   3       CH$WCHAR_A (.char, op);
1693    1767   2   END;
1694    1768   2
1695    1769   2 ! Start with the address and length of the record
1696    1770   2 !
1697    1771   2 buflen = .op - buf;
1698    1772   2 bufptr = buf;
1699    1773   2
1700    1774   2 ! Print the record.  We must allow for a segmented put if the size of the record is too big for the output f
1701    1775   2 !
1702    1776   2 DO
1703    1777   3   BEGIN
1704    1778   3   put (MINU (.buflen, .copy [copy$l_max_rec]), .bufptr);
1705    1779   3   buflen = .buflen - .copy [copy$l_max_rec];
1706    1780   3   bufptr = .bufptr + .copy [copy$l_max_rec];
1707    1781   3   END
1708    1782   2 UNTIL .buflen LEQ 0;
1709    1783   2
1710    1784   2 RETURN;
1711    1785   1 END;
```

```
                              07FC 00000        .ENTRY  COPY_TYPE_PRINT, Save R2,R3,R4,R5,R6,R7,R8,-;  1558
                                                        R9,R10
                  5E      F600  CE  9E 00002     MOVAB   -2560(SP), SP
                  50 00000000G  EF  D0 00007     MOVL    EXCH$A_GBL, R0                                1595
                  5A        04  A0  9E 0000E     MOVAB   4(R0), R10
                  56        08  AC  D0 00012     MOVL    REC, IP                                       1742
                  57            6E  9E 00016     MOVAB   BUF, OP                                       1743
                  59        04  AC  D0 00019     MOVL    LEN, COUNT                                    1745
```

```
                                    29  11 0001D            BRB      4$
                           50       86  9A 0001F  1$:       MOVZBL   (IP)+, CHAR
                           51  0000'CF40  9A 00022          MOVZBL   TABLE[CHAR], STRING
                                    19  13 00028            BEQL     2$
                           51  0000'CF41  9E 0002A          MOVAB    TABLE_BASE[STRING], STRING
                                    87  3C  90 00030        MOVB     #60, (OP)+
                           58       61  9A 00033            MOVZBL   (STRING), LEN
              67      01   A1       58  28 00036            MOVC3    LEN, 1(STRING), (OP)
                           57       58  C0 0003B            ADDL2    LEN, OP
                           67       3E  90 0003E            MOVB     #62, (OP)
                                    03  11 00041            BRB      3$
                           67       50  90 00043  2$:       MOVB     CHAR, (OP)
                           57       D6 00046  3$:           INCL     OP
                           D4       59  F4 00048  4$:       SOBGEQ   COUNT, 1$
                           50       6E  9E 0004B            MOVAB    BUF, R0
              53           57       50  C3 0004E            SUBL3    R0, OP, BUFLEN
                           54       6E  9E 00052            MOVAB    BUF, BUFPTR
                           54       DD 00055  5$:           PUSHL    BUFPTR
                           52       6A  D0 00057            MOVL     (R10), R2
                           53       DD 0005A               PUSHL    BUFLEN
              38      A2   6E       D1 0005C               CMPL     (SP), 56(R2)
                           04       1B 00060               BLEQU    6$
              6E      38   A2       D0 00062               MOVL     56(R2), (SP)
              FFOE   CF    02       FB 00066  6$:           CALLS    #2, PUT
                           53       38  A2  C2 0006B        SUBL2    56(R2), BUFLEN
                           54       38  A2  C0 0006F        ADDL2    56(R2), BUFPTR
                           53       D5 00073               TSTL     BUFLEN
                           DE       14 00075               BGTR     5$
                                    04 00077               RET
```

; Routine Size: 120 bytes.   Routine Base: EXCH$COPY_CODE + 0B8C

```
; 1713      1786 1 END
; 1714      1787 0 ELUDOM
```

.EXTRN LIB$SIGNAL, LIB$STOP

PSECT SUMMARY

Name	Bytes	Attributes				
EXCH$COPY_P'IT	813	NOVEC,NOWRT, RD ;	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
EXCH$COPY_CODE	3076	NOVEC,NOWRT, RD ;	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)

Library Statistics

File	-------- Symbols --------			Pages	Processing
	Total	Loaded	Percent	Mapped	Time
_255DUA28:[SYSLIB]LIB.L32;1	18619	22	0	1000	00:01.9
_255DUA28:[EXCHNG.OBJ]EXCLIB.L32;1	1151	147	12	79	00:01.4

COMMAND QUALIFIERS

;   BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:EXCCOPY/OBJ=OBJ$:EXCCOPY MSRC$:EXCCOPY/UPDATE=(ENH$:EXCCOPY)

```
; Size:         3076 code + 813 data bytes
; Run Time:        00:58.6
; Elapsed Time:    03:23.2
; Lines/CPU Min:   1830
; Lexemes/CPU-Min: 20110
; Memory Used: 367 pages
; Compilation Complete
```

EXCREQ
R32

MAILCVT
COM

SYSGTTSTR
MSG

EXCCOPY
LIS

USSLNK
COM

EXCDEFS
SDL

EXCLIB
B32

EXCCLDTBL
LIS

MAILUAF
COM

USSTSTLNK
COM

EXCHNG

XATEST
COM

EXCHANGE
MAP

LABIO
OPT

MSCPMOUNT
COM

LABIOCIN
OPT

EXCCMD
LIS

DRCOPY
PRM